The Anatomy of Gameplay: General Affect Prediction Across Games and Genres

L-Università ta' Malta
Institute of Digital Games

David Melhart
Institute of Digital Games
University of Malta

A thesis submitted for the degree of
Doctor of Philosophy

May, 2021

Abstract

General user modelling has long been the holy grail of many domains within artificial intelligence, including affective computing and games research. General affective user models in these fields could advance the understanding of human emotion and serve as a basis for numerous practical applications. However, as affective computing traditionally focuses on passive media, the transferability of affective models to dissimilar tasks is rarely observed. While psychophysiological signals used in affective computing can provide task-agnostic information, they do not scale well outside of a lab environment. In contrast, games research focuses on an inherently interactive medium that provides rich contextual information about the game-state and player behaviour. Unfortunately, most of the work in games user research focuses on game-specific applications, and the handful of studies on general affect modelling in the past have been limited by small, ad-hoc testbeds. Numerous research questions in the field still remain open, and it is clear that a more methodical approach is needed that examines more comprehensive datasets and different levels of generality.

To which degree can we predict player affect in unseen games? Is it possible to find transferable characteristics between dissimilar games and genres? This thesis asks these questions and attempts to answer them through a series of experiments on predicting player arousal. Along the way, it presents several contributions to affective computing and game research, including a robust pipeline for first-person affect annotation for interactive elicitors; a comprehensive online platform for data collection; the most extensive dataset to this date that contains affective labels for multiple games and genres; and an ordinal modelling pipeline using dynamic windowing, accounting for player memory. Studies in this thesis highlight the strengths of heuristic general features and present a thorough examination of the modelling of temporal dynamics of arousal through preference learning. The resulting models can transfer between games and genres with a high level of accuracy and the conclusions of this thesis point towards the robustness of time-related game-agnostic features in the modelling of games.

Acknowledgements

I am eternally grateful to my supervisors Georgios N. Yannakakis and Antonios Liapis for their trust and guidance. Meeting them quite literally changed my life and set me on my path. I am thankful for the years worth of opportunities that I got to learn and grow while working together with them. I am hopeful that our paths remain close in the future.

I would like to thank my friends and colleagues that I met at the Institute of Digital Games, who helped and motivated me throughout my journey. I would like to especially thank Daniele Gravina and Daniel Karavolos, who advised me during my studies and with whom I shared an office and many great memories; Jasper Schellekens, the beating heart of the Institute; Phil Lopes and Konstantinos Makantasis, who helped me understand much of the intricacies of the field; and Renata Ntelia, the best flatmate I ever had.

I am grateful to my family for all the support over the years. I would like to thank my parents Erika and András, my sister Réka, my aunt Éva, and Áron, who all helped me in times of peril one way or another.

Finally, I would like to express my gratitude to Birgit, my partner in crime. Without her eternal patience and encouragement, I couldn't have made it.

Statement of Originality

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified. (David Melhart)

To my grandparents, Magdi, Panni, and Karcsi,
who passed before I could see it through.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Towards the road to general Artificial Intelligence (AI) a vital stepping stone is the general recognition, manifestation, and simulation of emotion. While general intelligence and artificial psychology define two critical long-term goals of AI, many domains of AI research focus solely on general problem solving, knowledge transfer, or naturalistic language use. However, the intersection of intelligence and emotion would not just enable a better understanding of human cognition in general but also empower artificial systems to perform affect-based interactions across dissimilar settings. Even though affective states are often depicted on a circumplex plane with a neutral baseline in the middle (Russell, 1980), we humans are rarely experiencing absolute neutral emotions. Emotions permeate our everyday lives and interactions, and affective processes alter our basic cognitive functions and even our subsequent emotional appraisals (Prinz, 2004). Damasio's *Somatic-Marker Hypothesis* postulates that physiological processes connected to visceral affective reactions facilitate our behaviour and decision making at a fundamental level (Damasio, 1994). Therefore, future AI applications have to recognise and adapt to these emotional and affective states to reliably predict human action and behavioural patterns if they were to navigate our social landscape and interact with humans in a naturalistic way. Completing an affective loop through human-computer interaction (Togelius and Yannakakis, 2016; Yannakakis and Togelius, 2018), such an AI could potentially enhance different human experiences, increase learning capabilities, and carry out therapeutic healthcare tasks. To this end, general models of affect and emotion are needed, which perform well independently of domain and context.

It is hard, however, to build these models from the ground up out in the wild. As a more minor step towards general models of affect, this thesis focuses on the domain of videogames as a testbed for general affect modelling. Games have been captivating the human mind since the dawn of time, and with the advancement of technology, we now can realise self-contained virtual worlds with their own rules and boundaries. While these microcosms can facilitate a wide array of user experiences, they also provide an optimal test bed for affective computing applications. Even though games have explicit constraints, they are designed to support several affordances through different game mechanics. This bounded freedom makes them ideal for observing dynamic interactions and naturalistic emotion manifestation. Unfortunately, while videogames define the dominant application area for the study of general AI, little emphasis has been given to the ways general AI systems are possible in games beyond the task of gameplaying (Perez-Liebana et al., 2016a; Togelius and Yannakakis, 2016), including systems that create or even model player experience in

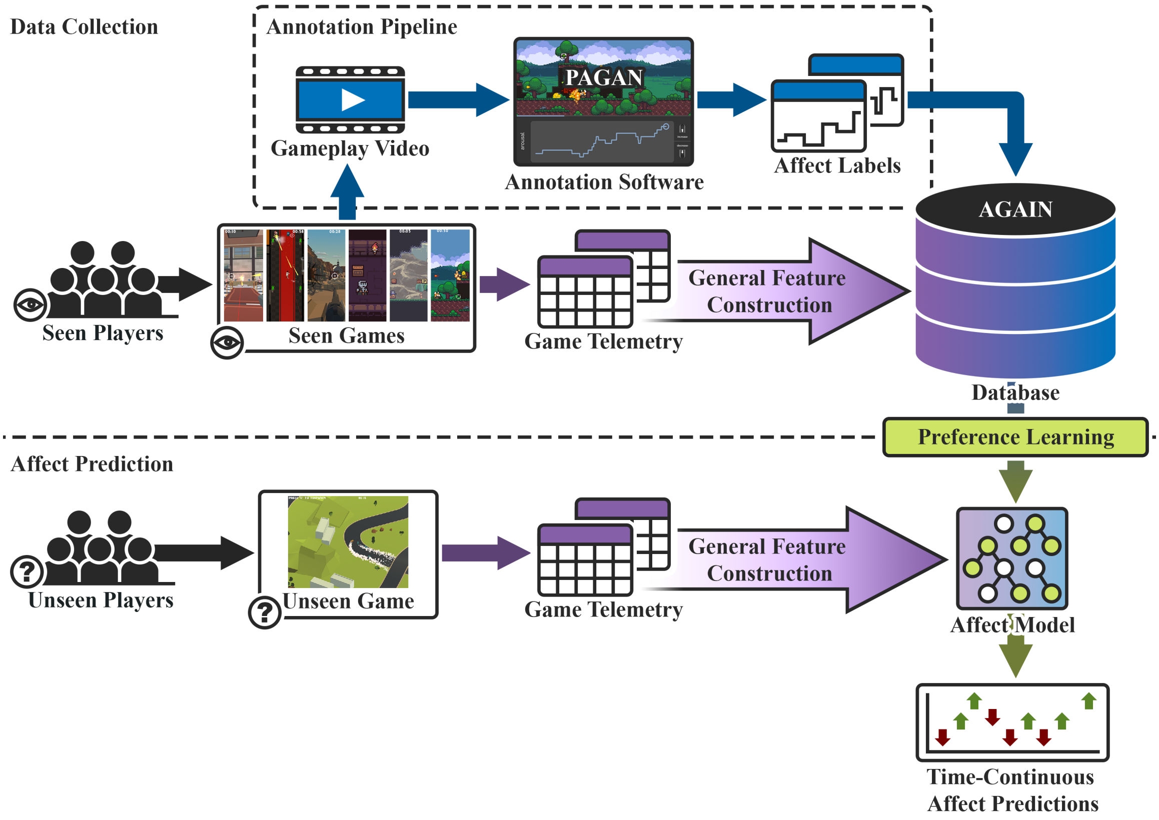


Figure 1.1: Illustration of the high-level general affect modelling pipeline presented in this thesis. The top of the figure shows the data collection procedure, which includes a robust first-person annotation pipeline for videogame play. The bottom of the figure shows the application of the constructed models on a previously unseen game and player base. While the overall framework is easily adaptable, the annotation software, PAGAN, and the compiled database, AGAIN, are specific to the thesis and introduced in Chapters 3 and 4, respectively.

a general fashion (Yannakakis and Togelius, 2018). Arguably, studying *general models of player experience*—which aims at predicting the experience of play in a game-independent way—is still in its infancy. The handful of examples in this vein are limited by ad-hoc testbeds and experience models built on small-scale game corpora (Shaker et al., 2015; Shaker and Abou-Zleikha, 2016; Camilleri et al., 2017).

1.1 Problem Formulation

Motivated by the lack of a comprehensive study on general player experience modelling, this thesis aims to fill this existing gap by creating an affective corpus for the purpose of general player modelling. As mentioned above, the focus on games is motivated by their unique structure; wherein games provide a well-bounded but interactive space. In contrast to traditional affective computing applications focusing on passive media such as reacting to video clips or sounds, games provide several points of interaction. As emotion manifestation in the wild is more dynamic and often depends on our interactions with our environment and other people in said environment, games present the perfect test bed for studying the temporal dynamics of emotions in a more naturalistic way. The central question which defines the investigation in this thesis is as follows:

> *To which degree can we predict player affect in unseen games?*

Answering this question, however, is far beyond the scope of this work. On the one hand, the scope of possible affective responses is broad, and on the other hand, there are too many possible paths towards predictive modelling to fit into one PhD thesis comprehensively.

Figure 1.1 illustrates the general approach of the presented work to affect prediction. This thesis applies a data-driven approach to predictive modelling and focuses on transferable characteristics of games. The work presented here assumes that there exist features of play that can transfer aspects of player experience across games and different game genres and that such features can be used to build accurate models of player experience in a general fashion. While in the above inquiry, affect represents a complex underlying visceral mental process that precedes all subsequent emotional responses, as a first step towards a comprehensive affective model of emotion in games, this thesis focuses on the *arousal* of players. Arousal has been chosen over other affective and emotional dimensions and categories as it carries high degrees of affective information within the domain of videogames. In games, arousal can describe the intensity of the play experience, which has further implications for the players' engagement, attention, enjoyment, and other positive psychological and cognitive outcomes. While some game research projects aim to measure these outcomes directly, this thesis focuses on the underlying arousal because *A)* it captures a visceral underlying dynamic of the player experience, and *B)* it ties the research more closely to affective computing. This is an essential criterion because while the goals of games user research and affective computing are often aligned, differences in the framing of research problems often make it hard to compare and build on shared results. This thesis aims to close some of this research gap and bring the two communities closer together.

The above inquiry undoubtedly necessitates a large and robust dataset with rich and diverse features capturing gameplay. As the goal is general modelling, this dataset should contain multiple games; these games should be similar enough to reasonably transfer models between them but still have key differences with a realistic distance between games. With the outline of this dataset, research presented here can be narrowed down to the context of this corpus and traditional machine-learning methods based on game telemetry.

1.2 Research Questions

Following the problem formulation above, two sets of research questions emerge. The first relates to general modelling, and the second relates to the pipelines used to gather, process, and model information about the players' arousal.

RQ1: Can *arousal* in games be reliably modelled in a general fashion?

RQ1.1: Can game-agnostic features perform comparably to specific features?

RQ1.2: Can models be transferred between games within a genre?

RQ1.3: Can models generalise over games from different genres?

RQ2: How to reliably capture and model *arousal* in games?

RQ2.1: How can we capture the first-person impression of arousal during gameplay?

RQ2.2: Can temporal biases be mitigated during data processing?

RQ2.3: To what extent can the temporal dynamics of the annotation be modelled?

The work presented here revolves around a large-scale dataset collected for this thesis, the *Affect Game AnnotatIoN* (AGAIN) dataset, presented in Chapter 4. Because most of the research into general modelling is based on ad-hoc datasets of a generally low number of games, the creation of this dataset was necessary to answer any of the research questions posed by this thesis. While creating the dataset posed some purely technical challenges in designing and developing the elicitor games, collecting the annotations for the dataset was not a trivial task either. To answer RQ2, data collection, processing, and modelling pipelines had to be devised. Firstly, to answer RQ2.1, Chapter 3 introduces the *Platform for Audiovisual General-purpose ANnotation* (PAGAN), which allows for the accessible, crowd-sourced collection of first-person annotations of arousal. Because PAGAN includes three different annotation methods for time-continuous annotation of multimedia content, a usability study helped narrow down the choice of annotation protocol for the collection of AGAIN. PAGAN served as a central platform to collect data seamlessly both in terms of game telemetry and annotations. The final pipeline is inspired by stimulated recall techniques and solves the conflict of interactive elicitors and first-person annotation by shifting the annotation task immediately after gameplay. As outlined by RQ1, experiments presented here focus on game-agnostic general features on different levels of generality. While the first experiments in Chaper 5 look at the robustness of game-based models built on general features—answering RQ1.1—latter experiments extend the scope to two levels of generality. Experiments in Chapter 6 attempt to answer RQ1.2 by looking at genre-based arousal modelling, while Chapter 7 searches for the answers to RQ1.3 by investigating general arousal modelling across genres. To answer RQ2.2 and RQ2.3 experiments in Chapters 5-7 implement a method of dynamic time-windowing and model arousal both in terms of relative change in the level of arousal and change in the gradient of the arousal trace, which describes the temporal dynamic (acceleration and deceleration) of the change in arousal.

1.3 Contributions

The thesis makes several contributions to games user research and affective computing in general. Beyond the experimental results, the work in this thesis also produced tools and a dataset, the scope and possible applications of which point far beyond this thesis. This section gives a short overview of the main contributions of the presented work and tools created for and during the PhD work, which contributed to the completion of this thesis. Projects in this thesis showcase preference learning for the general ordinal modelling of arousal. This focus is motivated by a growing body of literature in both game research and affective computing supporting the robustness and validity of ordinal modelling for human-generated data (Yannakakis et al., 2018).

1.3.1 Platform for Audiovisual General-purpose ANnotation

A major contribution of this thesis is the development of a new interface for affect annotation. The *Platform for Audiovisual General-purpose ANnotation* (PAGAN) was designed to enable first and third-person annotation of multimedia content in an accessible way. While developed partially to fill the gap left behind by outdated software packages, in the wake of the 2020 Coronavirus outbreak, crowd-sourcing became a necessity for the completion of the project. The platform features a highly configurable interface, both popular and new time-continuous labelling methods, and supports a crowd-sourcing solution for collecting large amounts of annotation. Beyond introducing the tool, a small companion study also showcases the accessibility of different annotation methods, highlighting the reliability of labelling tasks aimed at collecting ordinal data.

1.3.2 Affect Game AnnotatIoN Dataset

The other primary outcome of this thesis work is the construction of the *Affect Game AnnotatIoN* (AGAIN) dataset. The creation of the dataset was motivated by the lack of a comprehensive videogame database, which also includes affective annotations. The AGAIN dataset supports the research of general affect modelling with more than $1,100$ recorded gameplay from 9 games from 3 genres. While the database was designed to address the needs of the research presented here, the dataset also includes more than 37 hours of gameplay footage. Even though this video data is not used during modelling in the presented studies, it allows for future research involving deep-learning and computer vision. The dataset is made openly available for all research purposes.

1.3.3 General Arousal Modelling Through Preference Learning

The last main contribution of this thesis is demonstrating a robust pipeline for first-person annotation and modelling of arousal in videogames and showcasing the feasibility of general arousal modelling on different levels. Studies presented here showcase the robustness of heuristic general features derived from genre-specific game telemetry. Later projects focus on general modelling in the contexts of genre-based and genre-agnostic modelling. Results reveal the key features and processing methods, which lead to successful general models.

The presented research also outlines easy-to-adopt and cost-effective pipelines for the general ordinal modelling of arousal from pre-processing to feature construction and modelling. On the one hand, the presented research expands on existing pairwise learning methods with a dynamic windowing method, on the other hand, the presented research helped to shape —in part—the *Python Preference Learning Toolbox* (pyPLT). PyPLT is an all-in-one software solution for data processing and modelling for preference learning. Pipelines developed during this thesis work contributed to the development of pyPLT.

1.4 Publications

This section presents papers that have been published during the PhD studies. The first part of the section includes works that are directly contributing to this thesis. The second half of the section presents papers that are not part of this thesis work. During the PhD studies, 7 papers have been published at the time of writing, 1 is accepted for publication and 3 more are under review and submission. Out of these 11 papers, 2 are journal articles, and 7 are published in conference proceedings; parts of the thesis outcomes have also been published as a workshop and a demo paper. While all of the research published during the PhD revolves around applying affective computing techniques to games user research, some of the work is more exploratory that could fit neatly into this thesis work. Nevertheless, they contribute to the field of affective computing and games research and investigate how players appraise games in terms of their perceptions of computer agents, game aesthetics, and intrinsic motivation.

1.4.1 Part of the Thesis Work

Papers presented in this subsection are part of the thesis work. Works are listed alongside the chapter to which they are contributing.

1. Melhart, David, Konstantinos Sfikas, Giorgos Giannakakis, Antonios Liapis, and Georgios N. Yannakakis. "A study on affect model validity: Nominal vs ordinal labels." In *IJCAI Workshop on Artificial Intelligence in Affective Computing.* Proceedings of Machine Learning Research, 2020.

   While the paper does not contribute to a specific chapter or chapters, lessons learned here provide the backbone of the modelling tasks in this thesis work.

2. Melhart, David, Antonios Liapis, and Georgios N. Yannakakis. "PAGAN: Platform for Audiovisual General-purpose ANnotation." In *2019 8th International Conference on Affective Computing and Intelligent Interaction Workshops and Demos (ACIIW).* IEEE, 2019.

   This paper contributes to Chapter 3, with an overview of the PAGAN framework.

3. Melhart, David, Antonios Liapis, and Georgios N. Yannakakis. "PAGAN: Video affect annotation made easy." In *2019 8th International Conference on Affective Computing and Intelligent Interaction (ACII).* IEEE, 2019.

   This paper contributes to Chapter 2 and 3, with an overview of popular and contemporary annotation tools and a detailed introduction to PAGAN.

4. Camilleri, Elizabeth, Georgios N. Yannakakis, David Melhart, and Antonios Liapis. "PyPLT: Python Preference Learning Toolbox." In *2019 8th International Conference on Affective Computing and Intelligent Interaction (ACII).* IEEE, 2019.

   The paper contributes to the description of the toolbox in Chapter 2.

5. Melhart, David, Antonios Liapis, and Georgios N. Yannakakis. "The Affect Game AnnotatIoN (AGAIN) Dataset." In *IEEE Transactions on Affective Computing.* IEEE, under review.

   The paper contributes to Chapters 2 and 4, with an overview of affective corpora, and the description of the AGAIN dataset and a preliminary analysis of the data, respectively.

6. Melhart, David, Antonios Liapis, and Georgios N. Yannakakis. "Towards General Models of Player Experience: A Study Within Genres." In *2021 IEEE Conference on Games (CoG).* IEEE, 2021.

   The paper contributes to the pipeline described in Chapter 3 and the models presented in Chapter 5 and 6.

1.4.2 Outside of the Scope of the Thesis Work

Papers presented in this section describe additional exploratory studies done under the PhD studies but not included in this thesis work. As these works are not contributing to the thesis, their content is summarised below.

1. Melhart, David, Georgios N. Yannakakis, and Antonios Liapis. "I feel I feel you: A theory of mind experiment in games." in *KI-Künstliche Intelligenz 34(1)*, 2020.

   This paper on players' emotional *theory of mind* (Schaafsma et al., 2015) investigates how players appraise the emotions of game-playing agents based on the agent's behaviour and the player's performance and emotions. The study focuses on the perception of frustration as it is a prevalent affective experience in human-computer interaction. Results show that the player's observable emotions are not connected to the perceived frustration of the agent, suggesting that the subject's *theory of mind* is a cognitive process based on the gameplay context.

2. Melhart, David, et al. "Your gameplay says it all: modelling motivation in Tom Clancy's The Division." In *2019 IEEE Conference on Games (CoG).* IEEE, 2019.

   This paper attempts to find a computational mapping between gameplay data and aspects of player motivation in *Tom Clancy's The Division* (Ubisoft, 2016). The study was run in collaboration with *Ubisoft Massive Entertainment.* Experiments in this paper explore the degree to which such data can be a powerful predictor of players' survey responses. Core findings in this paper suggest that not only is it possible to infer the mapping between high-level gameplay metrics and survey-based annotations of complex emotional and cognitive states, but the inferred models have predictive capacities that reach certainty levels.

3. Prager, Raphael Patrick, Laura Troost, Simeon Brüggenjürgen, David Melhart, Georgios Yannakakis, and Mike Preuss. "An Experiment on Game Facet Combination." In *2019 IEEE Conference on Games (CoG).* IEEE, 2019.

   This paper examines the emotional impact of game facet combination in terms of matching and mismatching audio and visual facets. The study was run in collaboration with the *University of Muenster.* Results show that players generally prefer homogeneous facet combinations. These combinations also make more reliable elicitors, evidenced by higher accuracy of arousal models on these settings. It is also revealed that the audio facet of the elicitor plays a more prominent role in the robustness of predictive models of arousal. Finally, models trained on positive soundscapes are generalising better over other configurations.

4. Makantasis, Konstantinos, David Melhart, Antonios Liapis, and Georgios N. Yannakakis. "Privileged Information for Modelling Affect In The Wild." In *2021 9th International Conference on Affective Computing and Intelligent Interaction (ACII).* IEEE, under review.

   This paper investigates how affective models that are built in a controlled laboratory (in vitro) setting can be extended to real-world applications (in vivo) through the use of *privileged information.* The proposed privileged information framework is tested in a game arousal corpus that contains physiological signals in the form of heart rate and electro-dermal activity, game telemetry, and pixels of footage from two dissimilar games that are annotated with arousal traces. Arousal models are trained using all modalities (in vitro) and tested using pixels (in vivo) solely. Constructed models reach levels of accuracy obtained from models that fuse all modalities both for training and testing. The findings of this paper make a decisive step towards realising affect interaction in the wild.

5. Pacheco, Cristiana, David Melhart, Antonios Liapis, Georgios N. Yannakakis, and Diego Pérez-Liébana. "Trace It Like You Believe It: Towards Time-Continuous Believability Prediction." In *2021 9th International Conference on Affective Computing and Intelligent Interaction (ACII).* IEEE, under review.

   This paper presents a study on time-continuous non-player character believability prediction and introduces a new approach to assess the performance of ranking machine learning models. The novelty of this paper is treating believability as a time-continuous phenomenon. The research explores the suitability of two different affect annotation schemes for this assessment. Results suggest that a discrete annotation method leads to a more robust evaluation of the ground truth and subsequently better modelling performance.

   This paper also gives a minor contribution to Chapter 3 as it uses the same pipeline described in this chapter.

1.5 Thesis structure

The thesis is structured in the following way:

- **Chapter 2** gives an overview of the relevant literature of affective theories, player modelling, annotation frameworks, affective corpora, and preference learning. The section aims to provide a general theoretical background to ordinal affect modelling and contextualise the created tools and dataset in the fields of games user research and affective computing. The second half of the chapter is dedicated to the methods applied in this thesis and introduces a dynamic windowing method for pairwise preference learning.

- **Chapter 3** details the first-person gameplay and annotation pipeline for affective data collection in videogames as well as the interface of the PAGAN annotation framework. The chapter also includes the results of a small usability study tied to the development of PAGAN to ground the choice of the annotation tool used for collecting the AGAIN dataset.

- **Chapter 4** presents the AGAIN dataset. Sections in the chapter detail the elicitors and the process of data collection up to cleaning the dataset. The second half of the chapter is dedicated to an overview of the pre-processing of the dataset for machine learning and correlation analysis of the data.

- **Chapter 5** deals with game-based baselines for the latter general modelling. This chapter investigates the feasibility of representing a game through general features when contrasted to genre-specific ones. Additional analysis in the chapter explores the impact of the dynamic windowing method introduced in Chapter 2 on model performance.

- **Chapter 6** focuses on genre-based modelling. The three genres included in AGAIN are examined to see the robustness of general features against genre-specific ones in a more general scenario. The chapter investigates both the modelling of unseen games and augmenting game-based models with additional information from other games within the genre. Finally, the chapter continues investigating the impact of individual features and windowing methods on model performance.
- **Chapter 7** completes the experimentation work by looking at general arousal modelling in a genre- and game-independent way. In this chapter, previous models built on general features are reexamined in terms of general robustness. Additionally, two new scenarios are examined for genre- and game-agnostic modelling. In the first one, models are built on games from two genres and predict the outcomes of games in an unseen genre, while in the second one, experiments maximise the available information by building models based on eight games to test the unseen one. This chapter also concludes the investigation into feature importance and windowing methods started in Chapter 5.
- **Chapter 8** summarises the findings and main contributions of this thesis work, including practical contributions such as the tools and dataset created during the development of the thesis. The chapter also lists the limitations of the presented research and concludes with an outline for future research.

1.6 Summary

This chapter outlined the main research question and primary challenges of this thesis while drawing a clear roadmap to tackle them. The primary goal of the thesis is to find a way for game-independent general arousal modelling in videogames while bridging some of the research gaps between game user research and affective computing. To this end, this thesis proposes the collection and analysis of a large corpus of gameplay telemetry and affective annotation. A considerable portion of the work is dedicated to developing an annotation platform and interactive elicitors to collect this dataset. For modelling the collected data, this thesis relies on ordinal affect modelling through preference learning. The work focuses on this approach as evidence has shown that preference learning frameworks closely mirror cognitive and emotional processes, making them more robust and reliable for affect modelling tasks (Yannakakis et al., 2018). For the modelling phase, pipelines are finalised based on lessons learned while developing a comprehensive software package for preference learning applications. The presented machine learning projects examine representing and modelling gameplay through general features on three different levels: game, genre, and general. Experiments focus on modelling both the change in the level of arousal and the temporal dynamic of arousal. Subsequent analysis in each chapter investigates feature importance for successful modelling and the impact of processing the data through dynamic memory windows. Finally, this chapter showcased papers published during the PhD studies. While some of these papers contribute directly to this thesis, others are out of the scope of this investigation. The chapter concluded with an overview of the thesis structure.

The next chapter focuses on the related work and methodology used in the thesis. It introduces the theoretical framework of the studies presented here, contextualises the thesis in the fields of affective computing and game research, and gives an overview of popular and contemporary annotation tools and affective datasets. The latter half of the chapter focuses on preference learning and random forests used in subsequent chapters to solve ranking tasks.

Chapter 2

Background

A core challenge of affective computing is the investigation of *generality* in the ways emotions are elicited and manifested, in the annotation protocols designed, and ultimately in the affect models created. Affective computing research requires access to corpora containing affect responses and annotations across dissimilar tasks, participants and annotators to examine the degree to which general representations of affect are possible and meaningful. Traditional large-scale affective computing datasets feature affect annotation of static images, videos, sounds and speech files within a narrow context through which affect is elicited from a particular task. However, such datasets cannot advance research in general affective computing, as stimuli used to elicit affect tend to be very similar. Even when the various tasks under annotation may vary, those are still limited to a very *specific* context—such as viewing a set of social interactions under a theme or playing sessions of the same game.

While affective computing mainly focuses on elicitation through non-interactive means, the world is becoming more and more interactive by the minute. Recent years have pushed videogames and game research into the spotlight (Hamdy and King, 2017; Aranha et al., 2019), opening up new fields for the study of human-computer interaction, affective computing, and interactive emotion elicitation. Videogames are robust elicitors that provide an ideal test bed for affective computing applications: games are well-structured yet dynamic environments with clear cues, goals, and feedback. While the field of game research is growing rapidly, there are no easily accessible datasets available. Instead, most player modelling studies focus on either ad-hoc experiments (Martínez et al., 2011) or involve large industrial datasets (Bonometti et al., 2020) where the data is not accessible for a wider academic audience (Yannakakis and Togelius, 2018).

This thesis takes a structured approach to general affect modelling by reviewing past and contemporary annotation tools, affective datasets, and approaches to affect modelling. This chapter provides background on concepts used in or related to the topics mentioned above. Section 2.1 introduces the theoretical approaches to emotion representation and outlines related work in player modelling. This section also motivates subsequent studies on the dimensional representation of affect and contextualises the research in the field of games research. Section 2.2 presents past and contemporary annotation frameworks used in affective computing and the publicly available popular affective corpora they capture. This section contextualises the research in affective computing and provides the necessary background to Chapters 3 and 4. Finally, Section 2.3 discusses the methodological background for the preference learning models used in this thesis. While the section introduces the main paradigms in preference learning, the main focus of the section is on pairwise preference learning through pairwise transformation and classification. This section briefly discusses the *Python Preference Learning Toolbox* as well. Section 2.4 summarises the Chapter.

2.1 Theoretical Background

This section is dedicated to providing theoretical background on the affective state of arousal by presenting an overview of frameworks of emotion representation (Section 2.1.1) and player modelling in games research (Section 2.1.2), focusing on general modelling and an ordinal approach to games user research.

2.1.1 Theoretical Frameworks of Emotion

Theories of emotions are generally represented in two main ways: as *dimensions* or as *categories*. The former focuses on emotions as emerging sentiments, which are functions of simple affective dimensions (Schlosberg, 1954; Mehrabian, 1980; Russell, 1980). The latter promotes an understanding in which basic emotions are distinct from one another in function and manifestation (Ekman, 1992; Lazarus and Lazarus, 1996). Although efforts are being made to reconcile these two viewpoints (Cambria et al., 2012; Cowen and Keltner, 2017), most studies in the field of affective computing subscribe to either categorical or dimensional frameworks. Subsequent studies in this thesis use a dimensional representation of emotion. This section presents the aforementioned frameworks, highlighting their strengths and weaknesses, and motivates choosing a dimensional representation over a categorical one.

Categorical Representation of Emotions

Categorical emotion representation is largely inspired by the work of Ekman (1992). It is based on the assumption that humans elicit distinct emotions inherent to the human psyche and universally understood. Classical theories following in Ekman's footsteps generally focus on the six emotions of *anger*, *disgust*, *fear*, *happiness*, *sadness*, and *surprise* (Ekman, 1992). While subsequent research often expanded this view to over twenty emotional labels at times (Cowie and Cornelius, 2003; Cowen and Keltner, 2017), majority of the research still focuses on the six core emotions (Barrett et al., 2019). In contrast to affective computing, in videogame research, the focus of emotion detection is broader. While some studies do observe basic emotions (Jones and Sutherland, 2008; Roohi et al., 2018; Kalantarian et al., 2018; Yang et al., 2018), others focus on other emotional outcomes more generally associated with a game playing experience, e.g. flow (Balducci et al., 2017), boredom, engagement, and anxiety (Chanel et al., 2008, 2011; Balducci et al., 2017), frustration (Yang et al., 2018), engagement, challenge, and frustration (Asteriadis et al., 2012; Shaker et al., 2013), and fun (Shaker et al., 2010; Fortin-Côté et al., 2018). Although normative studies have confirmed the generality of emotional categories to an extent (Diehl-Schmid et al., 2007; Westbury et al., 2015), putting the theory into practice also brings about some conceptual limitations. On the one hand, the underlying assumption of a clear division between basic emotional responses is challenged by a criterion bias when categorising fuzzy responses (Aviezer et al., 2008). On the other, the focus of emotion classification on facial morphology is undermined by evidence pointing towards the subjective evaluation of emotions based on contextual cues and body movements (Barrett et al., 2011; Aviezer et al., 2012; Barrett et al., 2019). These issues present in games as well, often even magnified by the nature of games. Roohi et al. (2018) give a cautionary tale about facial recognition-based emotion detection; players often react contradictory to in-game stimuli (e.g. smiling when missing a jump).

Dimensional Representation of Emotions

Alternatively, emotions can be represented through orthogonal affective dimensions. In these frameworks emotions emerge from a two- or three-dimensional continuous space. Most models of this kind typically follow Russell's *Circumplex Model of Emotions* (Russell, 1980) or the *Pleasure-Arousal-Dominance model* (Mehrabian, 1980) and many contemporary annotation tools (Morris, 1995; Cowie et al., 2000, 2013; Lopes et al., 2017b) use one of these models for annotating one or more affective dimensions as well. Although other dimensional theories of affect exist—for example the tension-arousal-valence model of Schimmack and Grob (2000)—most of them recognise the goodness and the intensity of emotions as the two core dimensions (sometimes expanded with "control over a given emotion"). The aforementioned *PAD model* represents these aspects as *pleasure, arousal* and *dominance* (Mehrabian, 1980); as a simplified version, the *Circumplex Model* presents the same space on the *arousal-valance* axes (Russell, 1980). While the *Circumplex Model* is a popular framework (Yannakakis and Togelius, 2018), it is not without critique. As mentioned above, Schimmack and Grob propose a *tension-arousal-valence* model instead, which further differentiates between tension-relaxation and wokeness-tiredness in addition to valence (Schimmack and Grob, 2000). However, it is evident from the literature that *arousal* is one of the more consistent components of dimensional models of affect, representing the intensity of an experience. In that regard, arousal defines one of the most important aspects of player experience when it comes to videogames and videogame research. The intensity of the game facilities the play dynamics and the perceived challenge of the game (Klarkowski et al., 2016), which affects player engagement (Abbasi et al., 2019) and can lead to several psychological outcomes, including tension (Lopes et al., 2017a), frustration (Melhart, 2018), fun (Clerico et al., 2016), and flow (Seger and Potts, 2012). Hence, arousal can be viewed as a fundamental building block of player experience and is the core affect dimension included in the *The Affect Game AnnotatIoN Dataset* presented in Chapter 4. The main limitation of these frameworks is that they cannot describe complex and self-reflexive emotions without expert interpretation, reintroducing biases to the observations. However, this simplicity also results in high *face validity* (Nevo, 1985), reducing guesswork and *criterion bias* of the annotator (Martinez et al., 2014) even in the case of fuzzy responses, which otherwise would be hard to categorise.

2.1.2 Player Modelling

Player modelling is a field of games user research that specialises in understanding and simulating (Holmgård et al., 2014) videogame play through AI techniques (Yannakakis and Togelius, 2018). The two main areas within this field focus on profile aggregation and predictive modelling. Player profile aggregation usually relies on clustering algorithms and traditional user analytics (El-Nasr et al., 2016) to expose and visualise underlying patterns in player behaviour and experience. This line of research applies different forms of aggregation to create an abstract, higher-level representation of gameplay and player behaviour. Common computational approaches use *k*-means clustering, self-organising maps (Drachen et al., 2009), matrix factorisation (Lim et al., 2016), archetypal analysis (Bauckhage et al., 2015; Drachen et al., 2012), and sequence mining (Martínez and Yannakakis, 2011; Wallner, 2015; Makarovych et al., 2018). The primary aim of these studies is to acquire a static profile of general behavioural and psychological patterns which can describe the player population.

In contrast to player profiling, predictive modelling takes a more dynamic approach and aims to actively predict certain behaviours or emotional shifts of the player experience (Yannakakis and Togelius, 2018). This type of modelling often uses supervised learning and relies on gameplay data instead of aggregation and abstraction. Notable applications of player modelling include predicting player behaviour (Bakkes et al., 2012), such as churn (Runge et al., 2014; Periáñez et al., 2016; Viljanen et al., 2018), playtime (Mahlmann et al., 2010), or player experience (Makantasis et al., 2019; Yannakakis et al., 2018). Many studies in the past relied on low granularity category-based prediction (Asteriadis et al., 2012) and focusing on a gameplay session as one unit (Shaker et al., 2013; Shaker and Shaker, 2014); however, there has been a shift towards a more granular evaluation of moment-to-moment gameplay (Camilleri et al., 2017; Makantasis et al., 2019; Melhart et al., 2020a). This thesis follows this trend as well and focuses on the moment-to-moment prediction of arousal in games.

General Models in Games Research

While general gameplaying models have been investigated thoroughly thanks to popular benchmark environments such as the *Arcade Learning Environment* (Bellemare et al., 2013) or the *General Video Game AI Competition* (Perez-Liebana et al., 2016a), the same level of investigation is not observed in of other fields of game research such as content generation and user modelling (Togelius and Yannakakis, 2016). While some studies have been conducted on general modelling both in terms of behaviour (Vítek, 2020) and emotion (Camilleri et al., 2017) prediction, most studies either focus on testbeds with a single game (e.g. (Mahlmann et al., 2010; Periáñez et al., 2016; Viljanen et al., 2018; Makantasis et al., 2019)) or emphasise the generality of the *approach* rather than the generality of specific *models* (Liu et al., 2018; Snodgrass et al., 2019a; Makantasis et al., 2021). One possible reason for this slow adoption is the lack of comprehensive datasets. Even in research into creating general models that include multiple games, these studies predominantly use an ad-hoc selection of a few games. Shaker et al. (2015) investigated manual and automated feature mapping (Shaker and Abou-Zleikha, 2016) through transfer learning in just two games. More recently, Camilleri et al. (2017) looked at using game-agnostic top-down features for general affect modelling—such as *playtime* and encoded *valence* as *goal oriented* and *goal opposed*—across three games to model arousal. Makantasis et al. (2021) investigated a general approach to modelling arousal "from pixels and sounds to emotion" in four games. However, a general limitation of these studies is the ad-hoc set of test-bed games, which are often limited in scope or fall either too close or too far from each other. Studies into behavioural modelling show a similar ad-hoc focus. As a recent example, Vitek (2020) used three free-to-play games in their study into general churn prediction models. In one of the most comprehensive studies in the field, Bonometti et al. (2020) used *activity count* and *diversity* to abstract gameplay and model general engagement across six commercial games from a diverse set of genres. Unfortunately, the latter dataset is not available for open research use.

A central problem in general modelling—of any kind—is the definition of the models' input features. As different games have different mechanics and characteristics, models cannot rely on unprocessed telemetry. Models instead have to rely on a shared input space on which all features can be mapped across several games to solve this issue. The most transparent approach is to use some form of heuristic abstraction of features. Many studies, including this thesis work, opt for this approach as it provides a low-cost but robust way to define general features. In an industry scenario, game designers could create these general features or identify general elements during the design process. As the method relies on expert knowledge, it provides a good level of robustness and flexibility. Shaker et al. and Camilleri et al. showed the feasibility of this approach (Shaker et al., 2015; Camilleri et al., 2017), with mixed results. In studies by Shaker et al., the heuristic approach arguably showed more robustness compared to machine-constructed features (Shaker et al., 2015; Shaker and Abou-Zleikha, 2016). While this type of feature construction can rely solely on ad-hoc feature abstraction, formal systems can also help this process. One of the most popular of these formal systems is the Video Game Description Language (VGDL) (Schaul, 2013) which is used for the games featured in the General Video Game AI Competition (Perez-Liebana et al., 2016b). VGDL is a computer language that uses simple semantics to describe games in a very conscience and uniform manner. The language defines sprites, levels, interactions, and terminal states. The main limitation of VGDL is its simplicity. The language was designed to describe simple 2D arcade-style games but unfortunately cannot readily be applied to more complex games. Studies focusing on more complex games generally find features in similar categories to the sections VGDL encodes, with some additional contextual information (e.g. time and score) (Shaker et al., 2015; Camilleri et al., 2017; Bonometti et al., 2020). Another framework for generalising gameplay features is the *Player, Environment, Agents, System* (PEAS) framework by Snodgrass et al. (2019b). Beyond the titular categories, PEAS offers a hierarchical view of game design elements. Beyond game features, this framework also incorporates personality models into the input space. However, as a theoretical design framework for personalisation, PEAS focuses more on guiding the design process from abstract elements towards a concrete implementation (Snodgrass et al., 2019a). Additionally, studies can adapt psychological frameworks as well to abstract gameplay features. For example, using the Circumplex Model of Emotions (Russell, 1980) events can be classified as positive or negative valence and arousal or using the theoretical framework of *GameFlow* by Sweetser and Wyeth (2005) can help the identification of events that facilitate a flow experience. This type of encoding aims to capture some of the affective aspects of games and encode them already in the input space; however, it can be considerably more labour intensive and can introduce unforeseen biases into the data.

While finding general features for a game designer is an easier task, user experience models are often constructed by third party analytics teams who do not have access to the same design insight. Another route towards general models is applying machine learning to extract features from the games' telemetry. While supervised feature learning still requires some designer input in the form of feature labels, unsupervised feature learning provides a more hands-off approach. As an example, Shaker et al. (2016) used unsupervised transfer learning to map features from different games onto a shared input space. In that study, Principal Component Analysis (PCA) was applied to find a matching representation between games in terms of input dimensions. Shaker et al. applied *subspace alignment* (Fernando et al., 2013) on these projections to create a shared feature space. One weakness of this approach is that the mapping has to be learned and adjusted between the source and target domains. While heuristic features are ready to use with multiple sources, adapting and adjusting feature learning to multiple source domains at once is a more complicated task.

Finally, general models can be constructed based on completely game-agnostic features. One of the most popular approach is to use physiological or other biometric signals for this purpose (Martínez et al., 2011; Martinez et al., 2013; Fortin-Côté et al., 2018). This approach has the obvious limitation of collecting bio-signals, which are generally out of scope for commercial games. Additionally, studies have shown that ambiguities in facial expressions often lead to unreliable models for game playing (Roohi et al., 2018; Melhart et al., 2020c). Beyond peripheral signals, it is also possible to model player affect in an end-to-end manner using deep-learning techniques. Makantasis et al. showed the feasibility of this approach for affective computing by extracting general-purpose or game-agnostic features from pixels and sounds (Makantasis et al., 2019, 2021). While their results have not been tested as general models, Makantasis et al. (2021) showcased a robust method for creating general features in multiple games.

Towards Ordinal Affect Modelling in Games Research

Studies in both games user research and user experience research still predominantly use various types of absolute ratings both as part of statistical analysis (Phillips et al., 2018) or by treating the responses as outputs for a supervised learning method (Frommel et al., 2018). One of the most popular approaches is to collect Likert-like responses (Mekler et al., 2014) through general survey tools such as the Self-Assessment Manikin (Bradley and Lang, 1994) and the Intrinsic Motivation Inventory (Ryan et al., 1983), domain-specific questionnaires like the Hexad (Tondello et al., 2016), the Game Experience Questionnaire (IJsselsteijn et al., 2013), the Player Experience of Need Satisfaction survey (Ryan et al., 2006), and the Ubisoft Player Experience Questionnaire (Azadvar and Canossa, 2018) or ad-hoc questionnaires (e.g. Asteriadis et al. (2012); Yang et al. (2018)). Despite Likert-like responses being ordinal by nature (Yannakakis and Martínez, 2015; Yannakakis et al., 2018), they are often treated as either interval (Phillips et al., 2018) or nominal values (Frommel et al., 2018) nevertheless. The traditional data processing approach in the field of games user research is indeed interpreting survey answers as nominal labels and applying classification methods to infer the function between in-game behaviour, affective experience (e.g. measured by physiology), and the self-reported ground truth; for example, see Asteriadis et al. (2012); Kołakowska et al. (2013); Phillips et al. (2018); Frommel et al. (2018) among many. Due to this approach, annotations are often discretised on a lower level of granularity, with models predicting emotions on a game-session level.

While the focus on data aggregation and user profiles is still the dominant technique in the games industry (El-Nasr et al., 2016; Drachen and Connor, 2018), in contrast, the field of affective computing has been focusing more and more on continuous emotion representation and prediction, modelling moment-to-moment dynamic changes (Yannakakis et al., 2016). Although there have been recent shifts towards this approach in games research as well (e.g. Yang et al. (2018); Makantasis et al. (2019); Melhart et al. (2020a); Makantasis et al. (2021)), this approach has yet to see widespread adoption in the field. While many of these early studies put more emphasis on time-continuous predictions, a lot of them still work based on absolute ratings or apply classification (Periáñez et al., 2016; Yang et al., 2018; Makantasis et al., 2019), which introduces several limitations both on a technical and theoretical level (Yannakakis et al., 2017, 2018). Most of the issues arise from the discrepancy between a player's experience and the reported label through a rating scale. While ratings ask players to evaluate their experience in an absolute fashion, a growing body of evidence suggests that the human assessment of subjective value relies on relative comparisons which are based on anchor (or reference) points of earlier experiences (Damasio, 1994; Seymour and McClure, 2008; Yannakakis et al., 2018). Human cognition is indeed prone to a number temporal biases (Damasio, 1994)—including *anchoring* (Seymour and McClure, 2008), *habituation* (Solomon and Corbit, 1974), *adaptation* (Helson, 1964), and other *recency-effects* (Erk et al., 2003)—which go unaccounted for in absolute rating tasks. However, since these biases are still present, ratings become unreliable as there is no way to control for these effects. Instead, focusing on the relative differences as opposed to absolute judgements can lead to more reliable observations and more robust predictions (Yannakakis et al., 2018). This robustness is so strong that it holds up even through second-order processing, in which participant responses are recorded as absolute ratings but later converted into an ordinal representation, focusing on their relative relationships (Yannakakis et al., 2018; Melhart et al., 2020b). While absolute ratings can be converted to ordinal labels (Yannakakis et al., 2018), bounded scales come with their own limitations (Yannakakis and Martínez, 2015). Nevertheless, collecting pairwise comparisons through forced-choice surveys can be labour intensive (as the number of comparisons can grow quadratically when new options are introduced) and often leads to lower granularity data (Shaker et al., 2013; Shaker and Shaker, 2014). A good compromise is to collect unbounded ratings, which can still be interpreted in an ordinal fashion but preserve the relative relationship between data points (Yannakakis et al., 2017).

In the field of games user research, several papers contribute to a growing body of research proving the effectiveness of *ordinal affect modelling* which is aimed to capture these relative processes behind emotional responses (Yannakakis et al., 2018); see Martinez et al. (2014); Yannakakis and Martinez (2015); Yannakakis and Martínez (2015); Lotfian and Busso (2016); Melhart et al. (2020b); Yannakakis et al. (2018) among many. This approach evidently increases the inter-rater reliability and consistency of data annotations (Yannakakis and Martinez, 2015; Yannakakis and Martínez, 2015), and yields models which have a higher generality across affective corpora (Melhart et al., 2020b) and dissimilar videogames (Camilleri et al., 2017). While some of these studies rely solely on gameplay data (Melhart et al., 2019a; Prager et al., 2019), others focus on multimodal player data that fuse gameplay with physiological data (Martínez et al., 2011; Georges et al., 2018; Camilleri et al., 2017) or data from the video streams of players (Shaker et al., 2013). While these multimodal signals are often shown to increase modelling accuracy, collecting physiological data is currently too expensive and intrusive to be feasibly applied in large-scale industry studies. Most studies in ordinal affect modelling in game research apply *pairwise preference learning* (see Section 2.3.1) as it generally provides a simple but robust method to create ordinal models. Similarly, this thesis also applies preference learning to create general models.

2.2 Affect Annotation

This section discusses annotation frameworks and the affective corpora they capture in the field of affective computing.

2.2.1 Annotation Frameworks

In recent years, tools for affect labelling have diversified. Earlier examples such as FeelTrace (Cowie et al., 2000) (released in 2000), AffectButton (Broekens and Brinkman, 2013) (first released in 2009), and AffectRank (Yannakakis and Martinez, 2015) (released in 2015) aim to capture a complex phenomenon by measuring two or three affective dimensions at once, while tools like ANVIL (Kipp, 2001) or ELAN (Wittenburg et al., 2006) (released

Table 2.1: A Survey of Affect Annotation Frameworks. A table entry is indicated with 'UNK' if it is unknown.

Tool	Dimensions	Mode	Label	Installation	Oversight	Availability
FeelTrace (Cowie et al., 2000)	2 dimensions (arousal-valence)	time-continuous	bounded continuous circumplex	UNK	yes	Not available
ANVIL (Kipp, 2001)	Categorical labels	time-continuous	discrete labels	Standalone installer	yes	Available
ELAN (Wittenburg et al., 2006)	Categorical labels	time-continuous	discrete labels	Standalone installer	yes	Available
EmuJoy (Nagel et al., 2007)	2 dimensions (arousal-valence)	time-continuous	bounded continuous	Standalone installer (requires network configuration)	yes	Available
AffectButton (Broekens and Brinkman, 2013)	3 dimensions (pleasure-arousal-dominance)	discrete	bounded continuous	Standalone or online	no	Available
ANNEMO (Ringeval et al., 2013b)	1 dimension (configurable)	time-continuous	bounded continuous	Installation through Node.js	yes	Not supported
GTrace (Cowie et al., 2013)	1 dimension (negative to positive)	time-continuous	bounded continuous	Standalone installer	yes	Not available
CARMA (Girard, 2014)	1 dimension (negative to positive)	time-continuous	bounded discrete (configurable)	Installer (requires MATLAB[a])	yes	Available
AffectRank (Yannakakis and Martinez, 2015)	2 dimensions (arousal-valence)	time-continuous	discrete circumplex	Adaptation through PHP and JavaScript	no	Not supported
RankTrace (Lopes et al., 2017b)	1 dimension (tension)	time-continuous	unbounded continuous	C# source (pre-built version available; requires VLC[b])	no	Available
DARMA (Girard and Wright, 2018)	2 dimensions (configurable)	time-continuous	bounded continuous (optional circumplex)	Installer (requires MATLAB, VLC, and a joystick)	yes	Available
NOVA (Heimerl et al., 2019)	1 dimension or categorical (configurable)	time-continuous	bounded continuous or discrete labels	Standalone installer	yes	Available
RCEA (Zhang et al., 2020)	2 dimensions (arousal-valence)	time-continuous	bounded continuous circumplex	UNK	no	Not available
RCEA-360VR (Xue et al., 2021)	2 dimensions (arousal-valence)	time-continuous	bounded continuous	Python package	yes	Available
PAGAN (Chapter 3)	**1 dimension (configurable)**	**time-continuous**	**unbounded and bounded continuous and discrete binary (configurable)**	**No installation (online)**	**no**	**Available**

[a] `https://www.mathworks.com/products/matlab.html`

[b] `https://www.videolan.org/`

in 2001 and 2006 respectively) measure discrete categorical emotions. In contrast, from 2013 on-wards, studies focus increasingly more on one-dimensional labelling of continuous affect labels (Cowie et al., 2013; Girard, 2014; Lopes et al., 2017b; Melhart et al., 2019b). The shift away from multi-dimensional labelling can be explained by concerns over the increased cognitive load induced by these methods that comes with more complex tasks. Increased cognitive load can undermine the strengths of dimensional emotion representation (Cowie et al., 2013), as one emotional axis can take precedence over the other—which could impact face validity. Despite this shift, many annotation tools still rely on multi-dimensional annotation as evidenced by Table 2.1. While some modern annotation frameworks still offer discrete categorical labels, most of the tools out there feature a bounded continuous labelling method. Table 2.1 shows that the overwhelming majority of annotation tools focus on collecting time-continuous labels, with the odd one out being AffectButton (Broekens and Brinkman, 2013).

Two-dimensional tools are generally relying on the *Circumplex Model of Emotions* often restricting labels to a circular two-dimensional space. Many of these studies use a modified or customised version of FeelTrace (Cowie et al., 2000) (e.g. EmuJoy (Nagel et al., 2007), DARMA (Girard and Wright, 2018), CASE (Sharma et al., 2019), RCEA (Zhang et al., 2020)). Similarly, one-dimensional tools generally build on the GTrace annotation framework (Cowie et al., 2013). GTrace was created as a bounded, continuous annotation tool and quickly became popular for affect labelling in human-computer interaction and affective computing (Baveye et al., 2015a; Müller et al., 2015; Dellandréa et al., 2016; Dhamija and Boult, 2018). GTrace has limited memory and displays only the last few annotation values. Treating these bounded continuous annotation traces as interval data and processing them in an absolute fashion remains the prominent method of many studies (Gunes and Schuller, 2013; Metallinou and Narayanan, 2013). As this methodology necessitates that the trace is bounded to ensure a standard scale among raters, interval processing of annotation traces provides data in a form that can be analysed via a wide array of statistical and machine learning approaches. However, both one and two-dimensional tools which feature bounded tracing come with certain caveats. Since there is an upper and lower limit to the annotation scale, the evaluation becomes objective and absolute. Such absolute evaluations are prone to biases that primarily stem from inter-rater differences (Yannakakis and Martínez, 2015; Yannakakis et al., 2018). Supported by the *adaptation level theory* (Helson, 1964), *habituation* (Solomon and Corbit, 1974), the *somatic-marker hypothesis* (Damasio, 1994), and numerous studies within affective computing (Yannakakis et al., 2018) it appears that subjects experience stimuli in relation to their prior emotional and physiological states, experiences, and memories. Thus, any annotation task is subject to a number of *anchoring* (Seymour and McClure, 2008), *framing* (Tversky and Kahneman, 1981) and *recency* (Erk et al., 2003) effects. Nevertheless, absolute values are often attributed and compared between raters even if the given measurement framework defines them as ordinal (i.e. *Likert-scale* (Likert, 1932) and *Self-Assessment Manikin* (Morris, 1995)). Evidently, studies show time and time again that the evaluation of absolute scales is participant-dependent and does not suppose an underlying interval scale (Stevens et al., 1946; Ovadia, 2004; Yannakakis et al., 2018). Due to concerns regarding traces that are processed as intervals, *AffectRank* was introduced in 2015 as a real-time rank-based discrete labelling tool (Yannakakis and Martinez, 2015). As a two-dimensional annotation tool, the main improvement of AffectRank over FeelTrace was the focus on recording ordinal changes instead of absolute values. However, while AffectRank managed to preserve the ordinal nature of subjective ratings, the discrete scale sacrificed data resolution. As an answer to this issue, *RankTrace* was introduced in 2017 for unbounded and relative annotation (Lopes et al., 2017b). RankTrace caters to the subjects' relative judgement models as the annotator effectively draws a graph of their experience which also acts as the annotators' reference point. RankTrace produces continuous and unbounded traces which can be observed as ordinal changes and be processed in a relative manner (Lopes et al., 2017b; Camilleri et al., 2017; Melhart et al., 2020c). Similarly to the popular GTrace framework, the annotator sees their previous labelling history. Because RankTrace is unbounded, it lets the annotator react to the situation compared to previous experiences instead of forcing them to evaluate the stimuli in an absolute manner.

While there is a diverse set of tools out there, unfortunately, as evidenced by Table 2.1, some of them are not well supported, unavailable, or require researcher oversight. This limitation often restricts dataset sizes as experiments cannot be outsourced easily. Generally, less configurable tools need less attention from researchers, but they also require more planning when setting up experiments as participants will have to load elicitors and save labels on their own. A lesser but similar issue is the installation of annotation software. While most popular frameworks have standalone versions, they often have to be calibrated or even built from the source code if adapted (e.g. Yannakakis and Martinez (2015); Lopes et al. (2017b); Xue et al. (2021)). This further restricts the wide adaptation of these frameworks.

This thesis work presents the *Platform for Audiovisual General-purpose ANnotation* (PAGAN) to address the issues mentioned above in the field. PAGAN was developed as an easy-to-use tool that supports fast prototyping through an online interface (Melhart et al., 2019b). PAGAN implements three variations of one-dimensional affect labelling techniques, representing different methods for measuring the ground truth of affect: *GTrace*, as bounded and continuous; *BTrace* (binary trace), as real-time discrete; and *RankTrace*, as unbounded and continuous annotation techniques. While other contemporary tools offer automated labelling (Heimerl et al., 2019) and integration into mobile (Zhang et al., 2020) and virtual reality (Xue et al., 2021) environments, the strength of PAGAN lies in its flexibility and ability to collect a large amount of data online with relative ease (see Chapter 4 for more). The PAGAN framework is detailed in Chapter 3.

2.2.2 Affective Corpora

The availability of large-scale corpora comprising affect manifestations elicited through appropriate stimuli is a necessity for affect modelling. Therefore, creating datasets that are annotated with reliable affect information is instrumental to the field of affective computing at large. This section reviews representative affective corpora that rely on audiovisual elicitors and discuss the contribution of this thesis through the *Affect Game Annotation* (AGAIN) dataset to the current list of databases that are enriched with affect labels. While this section focuses on mapping out different affective corpora and positioning AGAIN in the field of affective computing, Chapter 4 presents about the collected data in detail.

Table 2.2 presents the outcome of this survey[1]. The section follows a systematic approach for reviewing the state of the art in affective corpora and examine the following factors that distinguish the surveyed datasets: the *mode*, *type* of the provided elicitors, the number of possible elicitor *items*, and overall size of the available *video* database (see second to fifth column of Table 2.2), the *number* of participants and their recorded *modalities* (see columns six and seven of Table 2.2), the annotation protocol in terms of the *mode* and *type* of the annotation (see columns eight and nine of Table 2.2), the affective *labels* (see column

[1] N/A indicates where the category is "not-applicable" (e.g. there are no participants when third-party videos are used) and UNK indicates if an attribute is "unknown".

Table 2.2: A Survey of Affective datasets of Audiovisual Content. 'N/A' means an entry is not available and 'UNK' means it is unknown.

	Elicitation				Participants		Annotation				
Database	**Mode**	**Type**	**Items**	**Video**	**Number**	**Modalities**	**Mode**	**Type**	**Labels**	**Annotators**	**Tasks**
MAHNOB-HCI (Lichtenauer and Soleymani, 2011)	Passive	Video	20 videos	20 hours	30	EEG, ECG, EDA, temp., resp., face and body video, gaze, audio	First-person	Discrete (9-step)	Arousal, valence, dominance, emotional keywords, predictability	30	20
DEAP (Koelstra et al., 2012)	Passive	Video	40 videos	40 mins	32	EEG, BVP, EDA, EMG, temp., resp., face video	First-person	Discrete (5-step)	Arousal, valence, dominance, liking, familiarity	32	40
DECAF (Abadi et al., 2015)	Passive	Audio, Video	40 audio 36 videos	1.5 hours	30	MEG, NIR face video, hEOG, ECG, tEMG	First-person	Discrete (5-step)	Arousal, valence, dominance	30	76
CP-QAE-I (Guntuku et al., 2015)	Passive	Video	144 videos	5 hours	N/A	N/A	First-person	Discrete (5-step)	Liking, Differential Emotions Scale	76	14

LIRIS-ACCEDE (Baveye et al., 2015b)	Passive	Video	9,800 videos	27 hours	N/A	N/A	First-person	Pairwise	Arousal, valence	1517(arousal)
										2442(valence)
											UNK

Aff-Wild (Zafeiriou et al., 2017)	Passive	Video	298 videos	30 hours	200	N/A	Third-person	Continuous bounded	Arousal, valence	44414	298
AffectNet (Mollahosseini et al., 2017)	Passive	Image	450,000 images	N/A	N/A	N/A	Third-person	Continuous bounded, categorical	Arousal, valence, 8 emotion categories	12	137,500
Sonancia (Lopes et al., 2017a)	Passive	Audio	1280 sounds	N/A	N/A	N/A	First-person	Pairwise	Arousal, valence, tension	UNK	10
SEWA DB (Kossaifi et al., 2019)	Passive, Active	Video	4 videos 1 task	27 hours 17 hours	398	Facial landmarks, FAU, hand and head gestures	Third-person	Continuous bounded	Arousal, valence (dis)liking intensity, agreement, mimicry	5	90
RECOLA (Ringeval et al., 2013b)	Active	Video	1 task	4 hours	46	ECG, EDA, face video, audio	Third-person	Continuous bounded	Arousal, valence	6	23
MazeBall (Yannakakis et al., 2010)	Active	Game	1 game	N/A	36	BVP(HRV), EDA, game telemetry	First-person	Pairwise	Fun, challenge, frustration, anxiety, boredom, excitement, relaxation	36	1
PED (Karpouzis et al., 2015)	Active	Game	1 game	6 hours	58	Gaze, head position, game telemetry	First-person	Discrete (5-step), pairwise	Engagement, frustration, challenge	58	1
DAG (Yang et al., 2018)	Active	Game	1 game	8 hours	58	ECG, EDA, EMG, resp., facial features, game video, and game events	First-person	Categorical and discrete	Arousal, valence, and 7 emotions	58	1
FUNii (Beaudoin-Gagnon et al., 2019)	Active	Game	2 games	N/A	190	ECG, EDA, gaze and head position, controller input	First-person	Continuous, discrete	Fun (cont.), fun, difficulty, workload, immersion, UX	190	2
AGAIN (Chapter 4)	**Active**	**Game**	**9 games**	**37 hours**	**124**	**Game video, game telemetry**	**First-person**	**Continuous unbounded**	**Arousal**	**124**	**9**

ten of Table 2.2), and finally the number of *annotators* and number of *tasks* each annotator had to complete (see the eleventh and twelfth column of Table 2.2).

It is apparent from Table 2.2 that affective datasets have gradually—over the last decade or so—drifted away from traditional induced elicitation and posed expressions and instead turned towards soliciting spontaneous emotion manifestations. Most of these datasets have focused mainly on affect elicitation through *passive* (i.e. non-interactive) audiovisual stimuli (see second row of Table 2.2). Passive audiovisual elicitors are a popular choice as they do not require any particular skill from the participants and are relatively easy to implement. In contrast, one can also find datasets that make use of *active* elicitors involving tasks in dyads and videogames—including RECOLA (Ringeval et al., 2013b) and player experience datasets such as PED (Karpouzis et al., 2015), DAG (Yang et al., 2018) or the FUNii Database (Beaudoin-Gagnon et al., 2019). These interactive tasks provide a more complex and multifaceted affective stimulus while organically structuring the participants' experience compared to passive elicitors.

Most affective computing databases surveyed (see tenth row of Table 2.2) capture affective dimensions such as *arousal* and *valence*, with some datasets offering labels for additional dimensions—such as *dominance*—and categorical labels. The surveyed datasets that have used games as affect elicitors—Mazeball (Yannakakis et al., 2010), PED (Karpouzis et al., 2015), and FUNii (Beaudoin-Gagnon et al., 2019)—tend to be less focused with regards to the labels used and instead aim to capture more complex game-related user states such as engagement, fun or challenge (except DAG (Yang et al., 2018)). This core difference makes such player experience datasets distinctive to affective computing primarily because any lessons learned on traditional affective databases are not directly applicable to player experience datasets and vice versa.

The affective datasets surveyed here appear to be rather split in terms of the annotation type used. While some (e.g. DEAP (Koelstra et al., 2012), MANHOB-HCI (Lichtenauer and Soleymani, 2011)) opt for self annotation (first-person), many databases (e.g. RECOLA (Ringeval et al., 2013b), SEWA (Kossaifi et al., 2019)) use only a few expert annotators in a third-person manner. There is a clear trade-off between these approaches. First-person annotations are ideal for capturing the subjective appraisal of emotional content, while third-person annotations are better at labelling emotion manifestation through inter-rater agreement (Afzal and Robinson, 2011).

The above systematic review of the literature highlights a lack of large-scale databases implementing an active elicitation mode, using multiple elicitor types and adopting a first-person annotation scheme. While datasets using passive elicitors are generally larger, the cost associated with using active elicitation limits these datasets. As Table 2.2 shows, the size of databases featuring active elicitors generally cannot reach the standard of datasets featuring passive elicitors. However, the passive elicitors of these datasets are also less diverse, generally limited to very similar annotation tasks. This does not advance research on general affect modelling, as researchers have to examine dissimilar datasets (Camilleri et al., 2017) that often comprise mismatching data collection methods and annotation tools or do not offer enough context variety (e.g. the FUNii Database (Beaudoin-Gagnon et al., 2019) features two similar games from the same franchise). AGAIN, the dataset collected for this thesis work addresses the aforementioned limitations by offering a large-scale corpus based on a set of dissimilar interactive affect elicitors annotated through a first-person protocol. While the dataset at the time of writing is limited to 9 games and their annotated arousal, the dataset is planned to be augmented through more affective dimensions and enriched through more games. The resulting dataset leverages the strength of active emotion elicitation while producing data in amounts comparable to databases featuring passive affect stimuli. Moreover, AGAIN provides a diverse database for general affect modelling research that is not possible within any of the existing corpora.

This thesis positions AGAIN at the intersection of traditional affective computing corpora and datasets focusing on player experience. Focusing on a core affect dimension (i.e. arousal) instead of a game-related complex emotional outcome aims to make affective computing research even more relevant for game user research and vice versa. As games are highly interactive media, the captured data and annotations encode player affect, behaviour, and game context. The focus on first-person annotations is motivated by a better capture of the emotional intricacies of gameplay. Finally, continuous unbounded traces are chosen as the annotation method of arousal using *RankTrace* (Lopes et al., 2017b) via the PAGAN online annotation framework (see Chapter 3). Such traces can be processed and machine-learned in several ways, including regression, classification and relational learning (Yannakakis et al., 2018).

2.3 Machine Learning Methods

This section details the data-processing and machine learning methods used in this thesis. Section 2.3.2 gives a short summary of the *Python Preference Learning Toolbox* as the development of this tool aided the refinement of the data processing and machine learning pipelines presented in this thesis.

2.3.1 Preference Learning

Preference Learning (PL) is a supervised learning paradigm in which an algorithm learns to distinguish between datapoints in an ordinal manner (Fürnkranz and Hüllermeier, 2011). The name of the method stems from early applications of PL, which involved recommender systems and modelled actual user preferences (Joachims, 2002). PL algorithms, however, are directly applicable to any supervised learning task in which the target outputs can be treated as ordinal data. PL has been shown to be more robust when applied to emotional data (Melhart et al., 2020b; Yannakakis and Togelius, 2018). The three main approaches to PL are the point-wise, pairwise, and list-wise methods (Liu, 2011). This section introduces all of these approaches along with their strengths and weaknesses.

Point-wise Preference Learning

In point-wise PL, the goal of the algorithm is to find a utility function $f(x)$, which returns higher scores for more preferred items. Formally if $x_i \succ x_j$ (x_i is preferred to x_j) then $f(x_i) > f(x_j)$. In practice, this method involves fitting a regression function to every (x_i, y_i) datapoint where $x \in X$ is the input and y is the corresponding label. In the point-wise case, the label y is generally an ordinal value, for example, the ordinal rank of item x in set X. Depending on how the problem is formulated, a point-wise ranking problem can be solved by many different algorithms, including regression, classification, and ordinal regression (Liu, 2011). All of these methods leverage the information already encoded by the label y regarding the relevance or ordinal ranking of the corresponding datapoint. Point-wise PL is the easiest to integrate into existing machine-learning pipelines as it relies heavily on existing techniques, and through the scoring function $f(x)$, it can evaluate unseen items separately. As they require less amount of data transformation and operate on one item at a time during the learning process, the time-complexity of point-wise PL problems can be considerably lower than pairwise and list-wise approaches. Unfortunately, point-wise PL considered sub-optimal as these algorithms are oblivious to the relative relationships in the data. As the algorithm learns on one item at a time, the position of the datapoint in a ranked order is invisible to the algorithm (Liu, 2011). In practice, it has been shown that other methods (such as pairwise PL) are more robust, providing more stable predictions, especially when the distribution of the labels is not normal or unknown (Fürnkranz and Hüllermeier, 2003; Melnikov et al., 2016).

Pairwise Preference Learning

In pairwise PL, the goal of the algorithm is to learn to rank items in the dataset based on pairwise comparisons. Some methods transform the data itself (Melhart et al., 2020b) (also see *pairwise transformation* in Section 2.3.3), while others leverage Siamese neural network architectures (Burges et al., 2005). The core of all these methods is the treatment of the output of the data, which essentially discards the labels but conserves the relationships they describe. The algorithm then learns to predict this relationship instead of any actual label of the data. While pairwise PL algorithms require pairwise training, they can still output ranking scores from unseen items independently. In the case of RankNet, LambdaRank, and alike, the algorithm's architecture already provides easy access to a point-wise scoring function (Burges et al., 2005; Burges, 2010). In the case of binary classifiers relying on data transformation for pairwise learning, getting the scoring function could be resource-intensive, but nevertheless possible via cumulative scoring (Fürnkranz and Hüllermeier, 2003; Melnikov et al., 2016; Melhart et al., 2020b). The drawback of this method is the increased time complexity of the task. Depending on the valid pairwise comparison the algorithm evaluates, the computational complexity of the task grows considerably. When a complete ranking is used, the complexity grows quadratically (Fürnkranz and Hüllermeier, 2003). Creating a point-wise scoring for items runs into the same conundrum, as calculating a cumulative score based on pairwise comparisons can be similarly taxing (Fürnkranz and Hüllermeier, 2003).

As briefly mentioned in Section 2.1.2, this thesis work focuses on *pairwise* PL. Most studies into ordinal affect modelling in games research follow this path. The two main approaches to pairwise PL are *first-order* and *second-order* processing of preferences. During first-order PL, player annotations are already captured as relative preferences. This method usually relies on forced-choice questionnaires and subsequently have less granularity. A good example of such studies is (Shaker et al., 2013), where Shaker et al. are using neuro-evolutionary algorithms to train models on first-order preference data. While this method produces remarkably reliable labels, a long line of research supports the validity of second-order processing as well (Yannakakis and Martínez, 2015; Yannakakis and Martinez, 2015; Lopes et al., 2017a; Camilleri et al., 2017; Melhart et al., 2020b; Yannakakis et al., 2018). During second-order PL, annotation data is captured as real-valued ratings. These ratings could be either bounded or unbounded, as subsequent processing discards their real value and conserves only the relative relationships they describe. The strength of second-order processing compared to first-order processing is the granularity and speed of the former. Ratings can be collected in real-time with ease and later transformed into relative rankings, supporting continuous affect modelling.

List-wise Preference Learning

In list-wise PL, the goal of the algorithm is to learn the optimal ordering of a list of items. List-wise PL is generally applied to information retrieval either by using information retrieval measures directly to evaluate models (e.g. AdaRank (Xu and Li, 2007))—instead of using pairwise metrics—or minimising a pairwise loss based on some unique relevancy metric (e.g. ListNet (Cao et al., 2007)). Generally, list-wise approaches were shown to be more robust (at least in the field of information retrieval) (Xu and Li, 2007; Cao et al., 2007) and the computational complexity of the task is generally lower compared to pairwise approaches. As an example, while the time complexity of RankNet is $O(m \cdot n^2)$[2], the complexity of ListNet is only $O(m \cdot n)$ (Cao et al., 2007). As finding an analogue to a relevance score outside of information retrieval applications is not a trivial task, most applications of list-wise PL algorithms remain in this field, although some examples do exist. For example, Wang et al. used list-wise PL to model the emotional content of documents by predicting the preferred nominal labels (Wang et al., 2011).

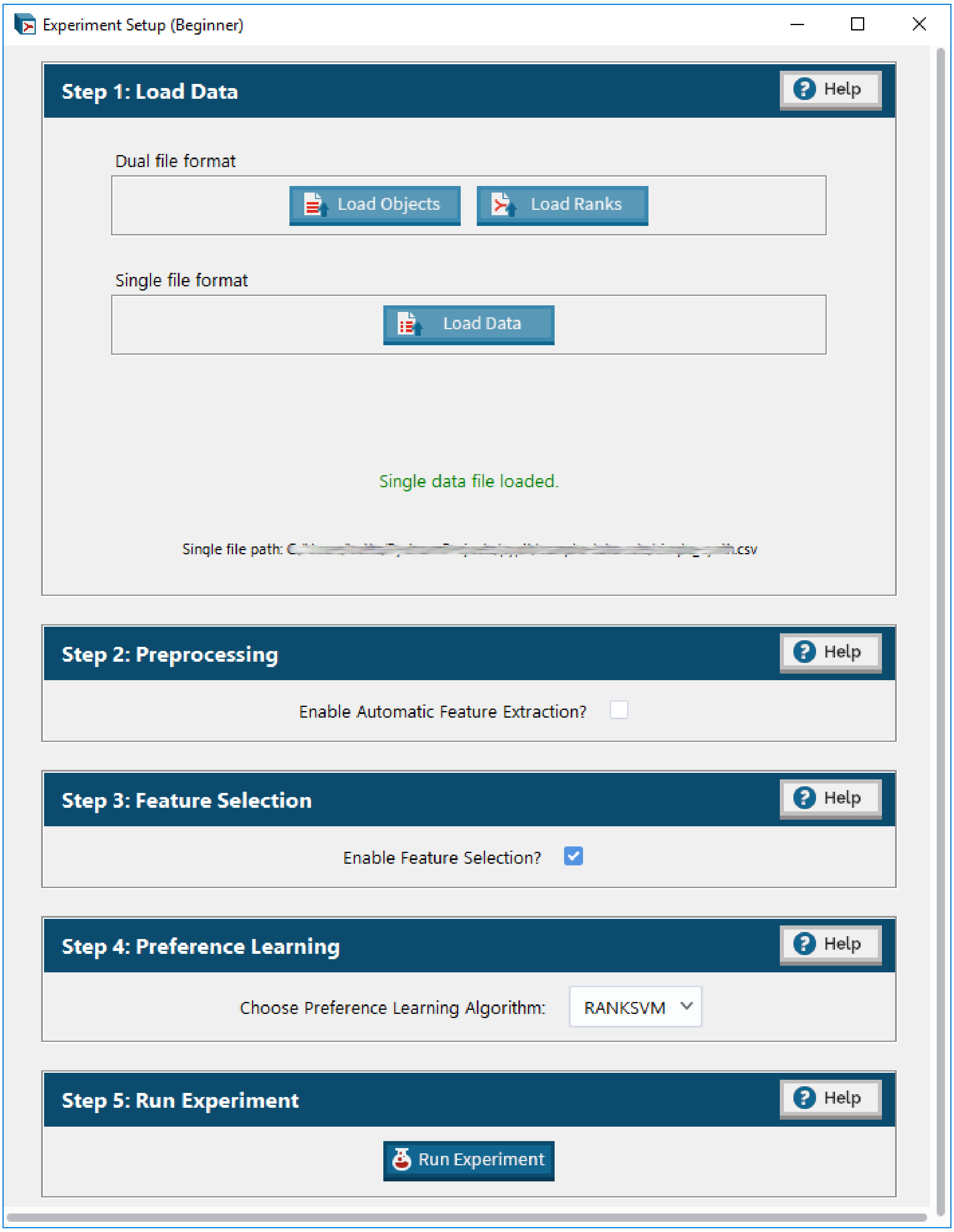
2.3.2 Python Preference Learning Toolbox

The *Python Preference Learning Toolbox* (PyPLT) is a toolset for *pairwise* PL. It was developed as an update to the outdated Preference Learning Toolbox written in Java (Farrugia et al., 2015). As Python became more and more popular in data science, the reliance on Java made the original toolbox less likely to be integrated into new projects. With much of the community shifting towards Python, new libraries are being developed and shared. This allows pyPLT to leverage well-constructed and well-maintained packages. However, as PL techniques often leveraging on other machine learning paradigms, even extensive Python libraries such as the Sci-Kit Learn toolbox[3] omit out-of-box PL pipelines. PyPLT is aimed to popularise and democratise ordinal label processing and PL. It is targeted to researchers and practitioners in affective computing, user modelling, preference handling, and human-computer interaction at large. The pyPLT toolbox may be used either as a software application via its Graphical User Interface (GUI) or as a library via its Application Programming Interface (API). Regardless of the interface used, pyPLT offers various methods for each step in the preference data modelling process.

PyPLT consists of a standalone application with a *graphical user interface* (GUI) and an underlying Python library. The GUI of pyPLT allows the user to select between two modules: *beginner* mode and *advanced* mode. Both modes provide a help dialog containing useful information to guide the user through using the application. While both modes of operation work based on the same shared Python library, beginner mode hides some of the more advanced features and implements automatic processes and best practices instead. Figure 2.1 shows the interface of the beginner module, which simplifies the experiment setup process into five quick and easy steps: *1)* loading the data set; *2)* automatic feature extraction using auto-encoders (Hinton and Zemel, 1994); *3)* automatic feature selection using a sequential selection algorithm; *4)* selection of a PL algorithm; and *5)* running the experiment. In the beginner module, the algorithms' parameters are either set to default values of the external libraries used (*TensorFlow*, *scikit-learn*, *Keras*), or where this is inapplicable, or the default value of the underlying framework is not optimal the parameters

[2] Where m denotes the number of queries and n denotes the number of documents per query in an information retrieval task.

[3] `https://scikit-learn.org/`



Step 1: Load Data

Dual file format

Single file format

Single data file loaded.

Single file path: C[redacted].csv

Step 2: Preprocessing

Enable Automatic Feature Extraction? □

Step 3: Feature Selection

Enable Feature Selection? ☑

Step 4: Preference Learning

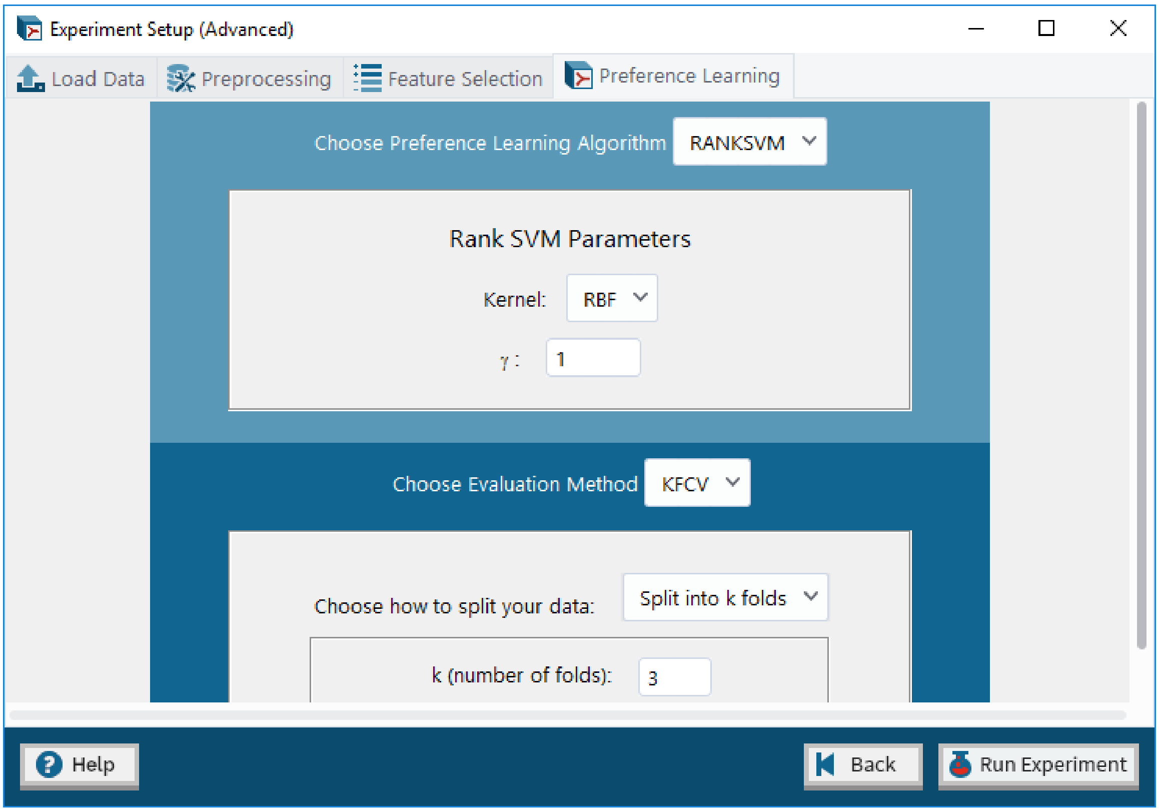
Choose Preference Learning Algorithm: RANKSVM

Step 5: Run Experiment

Figure 2.1: Screenshot of the beginner mode which hides the more detailed options available in the advanced mode, offering a more simplified interface for setting up an experiment.

follow the earlier version of PLT (Farrugia et al., 2015).

The advanced module follows the same five steps as the beginner mode but expands the



Choose Preference Learning Algorithm: RANKSVM

Rank SVM Parameters

Kernel: RBF

γ : 1

Choose Evaluation Method: KFCV

Choose how to split your data: Split into k folds

k (number of folds): 3

Figure 2.2: Screenshot of the PL tab (advanced mode) which offers three different algorithms and evaluation methods.

number of options by extending each of the first four steps into its own detailed tab (e.g., the preference learning tab shown in Figure 2.2). Each of these tabs contains options and parameters through which the user may fine-tune the experiment setup. The GUI includes three algorithms to solve the PL task. The first one is a ranking Support Vector Machine (rankSVM) based on the original implementation of ranking SVM in the Java PLT (Farrugia et al., 2015), which leverages a one-class SVM. In general, SVMs work by using a predefined kernel function to map the data instances onto geometric points in a high-dimensional space according to the input features that define them (Cortes and Vapnik, 1995). The second one is a Feedforward Neural Network with Backpropagation, which implements an artificial neural network (ANN) with gradient-descent. This method adjusts an ANN's weights iteratively to minimise the error between the predicted network output and the desired pairwise preferences over the given data instances. The error is calculated using the Rank Margin function. Finally, the third one is RankNet (Burges et al., 2005), which is an extension of the ANN backpropagation algorithm that uses a probabilistic cost function to handle ordered pairs of data. RankNet uses the binary cross-entropy function (Burges et al., 2005) as the error function. Additionally to these algorithms, the pyPLT library includes another ranking SVM, which uses pairwise transformation to infer information about the preference relations in the dataset. Section 2.3 introduces this method in detail. When running an experiment via the GUI, a progress bar and training report are displayed during execution. Upon completion, a report with the experiment's details and its results is shown to the user. Users can also save the experiment report and the model (for a given data fold,

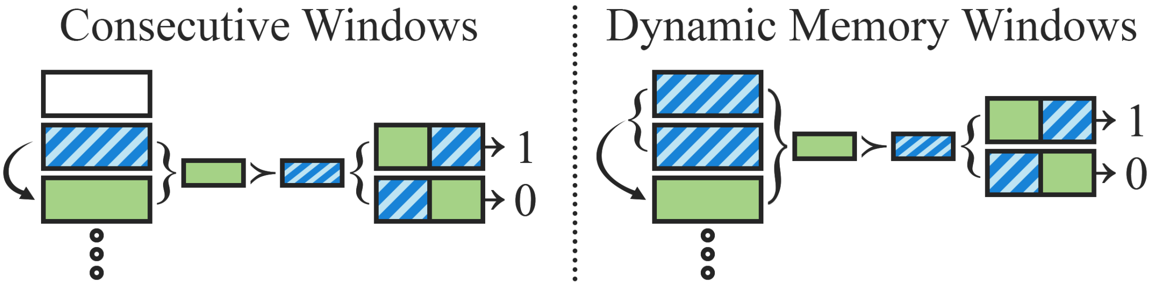


Figure 2.3: Difference between the pairwise transformation with consecutive windows and dynamic memory windows. In both instances the third datapoint (green) is compared to the previous data (blue striped).

if applicable) to a human-readable CSV file, either via the GUI or via the API.

2.3.3 Ranking through Random Forests Classifiers

This section presents the implementation of a Ranking Random Forest, which is used in subsequent studies in this thesis (see Chapters 5-7). Models in this thesis rely on random forests because they are simple yet robust algorithms that allow for fast training. The inherent properties of the algorithm also allow for a reliable assessment of feature importance. The work presented here relies on a method of *pairwise transformation* with which any labelled dataset can be transformed into a representation suitable for PL. This procedure reformulates the PL task in a way that any binary classifier can solve it.

Pairwise Transformation

Pairwise PL as introduced in Section 2.3.1, is a supervised learning paradigm in which an algorithm learns to distinguish between datapoints in an ordinal manner (Fürnkranz and Hüllermeier, 2011) through pairwise comparisons. Formally, through a *pairwise transformation* for every pair of data point $(x, x') \in X$ and label $(\lambda_x, \lambda_{x'}) \in L$ two new points $(x - x')$ and $(x' - x)$ and two new labels, y and y' are created. In case of $\lambda_x \succ_{P_t} \lambda_{x'}$ (x is preferred to x') $y = 1$ is assigned to $(x - x')$ and $y' = 0$ to $(x' - x)$, indicating the direction of the preference relation. P_t is a *preference threshold* parameter (between 0 and 1) that controls the required difference between two labels to be considered as a preference. The resulting dataset can grow quadratically in size depending on the valid preference relations in the dataset. The pairwise transformation is applied to each query, i.e. within each play session separately. The reformulated problem can be solved by any binary classifier, and due to keeping two observations per pairs, the baseline of the transformed dataset is always 50%.

Dynamic Memory Windows During Pairwise Transformation

While the pairwise transformation is often applied in an all-against-all strategy (Melhart et al., 2019a, 2020c), in time-continuous data, it makes sense to preserve the continuity of the data and only compare adjacent datapoints sequentially. At its basic level, this method only cares about the proximal changes from one moment to the next. While increasing the memory window by comparing more than two datapoints again in an all-against-all fashion has been proposed before (Camilleri et al., 2017), this method is simply producing a subset of a dataset-wise all-against-all transformation.

Instead, this thesis work introduces a simple but effective way to transform the dataset in such a way that the new datapoints encode the time-related dynamics of the original dataset. To achieve this, a dynamic windowing method is applied where within a session at each time-step, the current datapoint is compared not to the previous datapoint, but all previous datapoints combined (see Figure 2.3). This method compares new points to a smooth session history at each time step. In this thesis work, when this technique is applied, the past window size is always set to include all available previous datapoints in the session. This means that at time-step 3, this datapoint is compared to time-steps 1 and 2 combined, and at time-step 5, this datapoint is being compared to time-steps 1 to 4 combined. While a shorter window is possible, a preliminary exploration showed that larger windows generally worked better. Although this implementation of information retention from previous episodes is somewhat naive, unlike other algorithm-dependent pairwise PL methods, the pairwise transformation with dynamic memory windows is algorithm-agnostic. Hence, the parameter can be applied alongside any classifier.

Random Forrest Classifiers

This thesis uses *Random Forest* (RF) classifiers for PL. As has been briefly mentioned in the intro to this section, a preliminary analysis showed the robustness of this algorithm compared to ranking SVMs, Multi-layer Perceptrons, and RankNet for the specific task at hand. RFs are widely used algorithms in affective computing due to their fast training and robustness (Chatzakou et al., 2017; Hazer-Rau et al., 2020) often on par with deep learning methods on certain tasks (Kratzwald et al., 2018). An RF is an ensemble learning method used for classification and regression. As the name suggests, RFs operate by constructing a multitude of randomly initialised independent decision trees during training and use the *mode* of their predictions as a meta output. Decision trees themselves are simple yet powerful machine learning algorithms for predictive modelling; they operate by constructing an acyclical network of nodes, which split the features of the given dataset into simpler decisions (Lewis, 2000). For experiments presented in this thesis *Scikit-learn* Python library (Pedregosa et al., 2011) is used. *Scikit-learn* implements decision trees through an optimised Classification And Regression Tree (CART) algorithm first proposed by Breiman (Breiman, 2017). The CART method uses a generalisation of the binomial variance to evaluate the impurity (and thus splitting criterion) of nodes (Loh, 2011). It also relies on a process of "overgrowing" and pruning trees (Lewis, 2000) to minimise training errors without overfitting.

While RFs show surprising robustness, especially in the face of noisy data, another benefit of using RFs is the possibility to calculate the feature importance of models to explain the machine learning process better and understand both the data and the predictive results. In this thesis, feature importance is calculated as the *Mean Decrease Impurity* (MDI) (Louppe et al., 2013), which measures the average amount by which a feature decreases the weighted impurity across all trees in the forest. The MDI value is normalised between 1 and 0, the latter meaning the feature is irrelevant. The ordinal importance of the features can be observed by ranking them by their corresponding MDI values.

2.4 Summary

This chapter provided an overview of core concepts in emotion representation and player modelling. Section 2.1 on the theoretical background of the thesis introduced affective frameworks, emphasising the strengths of a dimensional understanding of affect when the goal is continuous modelling of changes in fuzzy emotional states; general modelling in games research, and ordinal affect modelling. Section 2.2 detailed contemporary annotation tools and affective corpora, providing background for the following Chapters (3 and 4). These sections also contextualise the thesis in the fields of game research and affective computing. Finally, Section 2.3 gave an overview on preference learning, introduced the Python Preference Learning Toolbox (pyPLT), and detailed the methodology behind the work presented in this thesis. The thesis follows a pairwise approach to preference learning. While pyPLT is not contributing directly to the modelling tasks in this thesis, work on this project helped crystallise the machine-learning pipeline used in subsequent studies. The section explained this pipeline on a high level, introducing the techniques of *pairwise transformation*, *dynamic memory windowing*, and gave a brief overview on random forest classifiers that will be used throughout this thesis.

The next chapter introduces the first-person affect annotation pipeline proposed by this thesis and the annotation platform developed to realise it. The next chapter also presents a usability study that evaluates the aforementioned annotation tool and the labelling methods implemented within it.

Chapter 3

First-Person Affect Annotation in Videogames

The question of how to reliably collect affect annotations does not have a trivial answer. Given the ever-increasing use of data-hungry affect modelling techniques, the need for larger and more reliable affective corpora is growing. While video-based annotation—a popular method in affective computing—is already a costly and cumbersome process, moving from passive to interactive elicitors has other caveats as well. The primary challenge with these types of elicitors—such as videogames—is that participants cannot both act and annotate simultaneously. Affective computing datasets recording the emotional outcomes of videogames have to devise pipelines where the gameplay and annotation tasks are shifted to collect reliable labels. Ideally, the annotation process should be seamless and present the least amount of cognitive load for participants. Unfortunately, most popular annotation tools out there—including FeelTrace (Cowie et al., 2000), ANNEMO (Ringeval et al., 2013a), AffectButton (Broekens and Brinkman, 2013), GTrace (Cowie et al., 2013), CARMA (Girard, 2014), AffectRank (Yannakakis and Martinez, 2015), and RankTrace (Lopes et al., 2017b)—require installation, calibration, and either trained expert annotators or constant researcher oversight. This limited accessibility leads to sparse affective corpora both in use and size. This limitation is even more severe for newly emerging fields, without large established corpora—such as game user research—where data collection is necessary.

To address these limitations, democratise data collection, and make ordinal modelling more accessible, developing a new pipeline for first-person affect annotation in videogames was necessary. To support this pipeline a tool for *general-purpose*, *online* video labelling was designed, the *Platform for Audiovisual General-purpose ANnotation* (PAGAN). PAGAN is publicly available at `http://pagan.institutedigitalgames.com/` and free to use for research purposes, and it provides researchers with an easy and *accessible* way to crowdsource affect annotations for their videos. In contrast to other popular annotation tools, the core functionalities of PAGAN do not require a local installation and are designed to help researchers organise and disseminate their research projects to many participants. Inspired by McDuff et al. (2013), the whole annotation process is done through a web interface operating on any modern web browser. Outsourcing the labelling task is as simple as sharing the corresponding project link.

PAGAN currently features three one-dimensional affect labelling techniques representing different methods for measuring the ground truth of affect: *GTrace* (Cowie et al., 2013), *BTrace* (a modified version of AffectRank (Yannakakis and Martinez, 2015)) and *RankTrace*

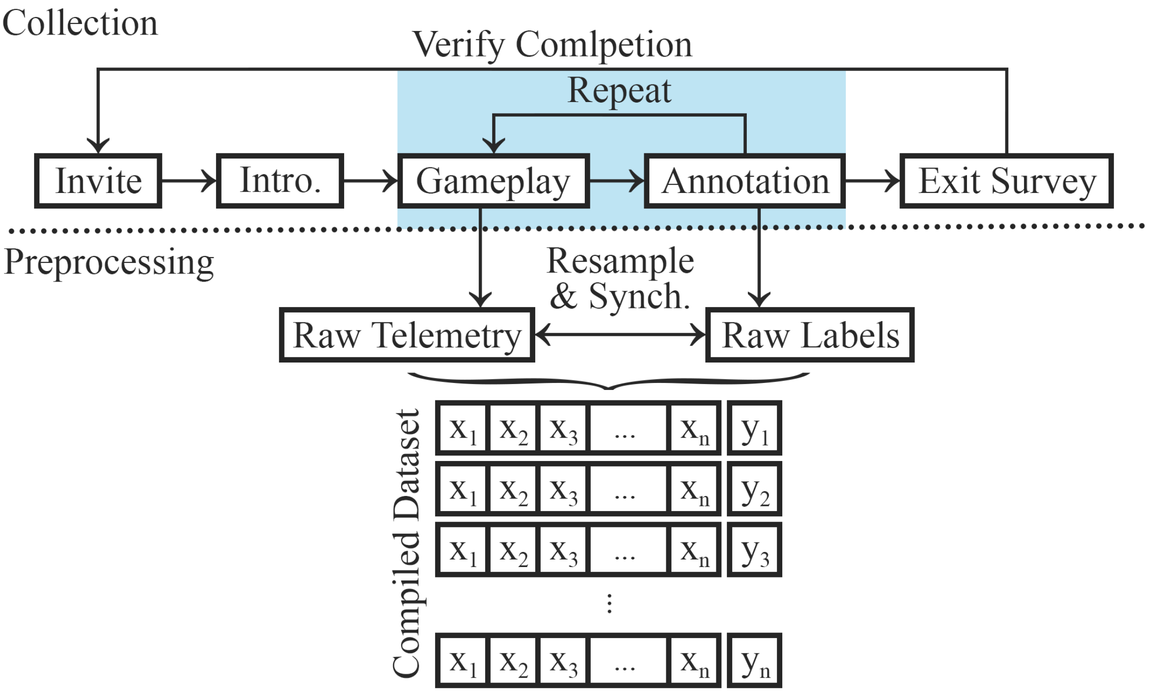


Figure 3.1: Data collection and annotation pipeline for videogames enabled by PAGAN. The central highlighted block of gameplay and annotation can be repeated multiple times. With the exit survey a verification code can be set up to allow integration with popular crowd-sourcing services.

(Lopes et al., 2017b). For advanced users with more familiarity with web technologies, an open-source repository is available[1] through which they can further customise the tool and the collection method. Finally, the repository includes a module that allows for the integration with games built-in Unity[2].

Section 3.1 discusses a proposed pipeline for the first-person collection of gameplay telemetry and affect annotation. This section also sets up Chapter 4, as the same methods were used to create the presented dataset as well. Section 3.2 introduces the *Platform for Audiovisual General-purpose ANnotation* (PAGAN). PAGAN was developed to allow for a large-scale crowdsourced collection of affective labels. While the tool was specifically designed with this thesis in mind, it has been released as an open-source tool (Melhart et al., 2019b), and to date, PAGAN has over 30 registered users. Section 3.3 presents a short study on the usability of the framework, which determines the annotation method selected for future studies. Finally, Section 3.4 summarises the chapter.

3.1 First-Person Affect Annotation Pipeline

First- and third-person annotation in affective computing refers to the viewpoint of the annotator from the perspective of the annotated subject (Yannakakis and Togelius, 2018). Participants labelling their own emotions are considered first-person annotators, while if an outside observer—usually an expert—is labelling the participants' emotions are considered

[1] `https://github.com/davidmelhart/PAGAN`

[2] `https://unity.com/`

third-person annotators. In the former case, the first-person annotation has high levels of internal reliability as the participants are trusted to know about their own emotions and label them in a self-consistent way. In the latter case, the third-person annotation has a high level of external validity as expert annotators can be trusted to evaluate different participants similarly. Repeated tests with third-person annotators can also increase the inter-rater reliability of labels. While this view covers the participant-annotator relationship, it does not consider the participants' connection to the elicitor. This is because it is a common practice in affective computing to collect annotations in reaction to a passive elicitor (see the review of affective corpora in Section 2.2.2). These elicitors are generally created by a third person, and the annotator has no agency in how they are configured. This setup allows for repeated first-person annotations on the same elicitors, which can be combined into a more stable ground truth later (Kächele et al., 2016).

However, the same experimental setup cannot be applied to interactive elicitors. Here, first-person annotators are labelling their own emotions and shaping how the elicitor is configured. In the case of videogames, each gameplay recording becomes a unique video as players take different paths through the virtual space (this is true to even simple, linear games). This unique setup comes with its own caveats. Firstly, first-person annotations cannot be collapsed anymore into a "gold standard" as there are no stable elicitors between participants. Secondly, as participants are preoccupied with playing, they cannot also label their experience; participants cannot generate both the elicitor and the annotation simultaneously. While the former problem could be sidestepped using third-person annotators on the recorded gameplay videos, this results in a very different measurement. Collecting annotations in this manner decouples the experience and the labelling task as annotators no longer annotating the emotional content of an event they experienced but their own interpretation of the context and the emotion manifestation of a third party. This thesis instead posits that first-person annotations could be used reliably through the adaptation of stimulated recall methods (Lankoski and Björk, 2015). This also offers the solution to the latter problem outlined above by shifting the annotation task right after the game playing task.

This thesis implements this pipeline through repeating gameplay and annotation blocks, as illustrated by Figure 3.1. The setup leverages the PAGAN framework for capturing gameplay videos and guiding the process. Gameplay videos are automatically recorded then immediately played back to the player to minimise the loss of episodic memory about the elicitor and the effect of semantic memory on the annotation task. In the complete data collection cycle, participants are invited through PAGAN to an experiment. After an introduction to the task, the system forwards the participants to a game. Games produced in Unity communicate with the platform to initialise recording sessions and start and stop video recordings. The telemetry logging is implemented independently on the game side; however, PAGAN handles the broadcast telemetry as well. This ensures that the system uses the same unique identifiers to store the raw telemetry submitted to the database as the annotation module, making the synchronisation of the two signals trivial. After the game playing task has finished, the recorded gameplay footage is submitted to the PAGAN platform, automatically creating a new entry for annotation. From here, PAGAN loads up the annotation task as described in Section 3.2.2. This block of gameplay-annotation can be repeated as many times as needed. If the data collection is crowdsourced, a completion code can be given out at the end of an exit survey to verify the completion of the experiment. This pipeline of data collection has already been used for the AGAIN dataset and for a study on the time-continuous prediction of non-player character believability (Pacheco et al., in

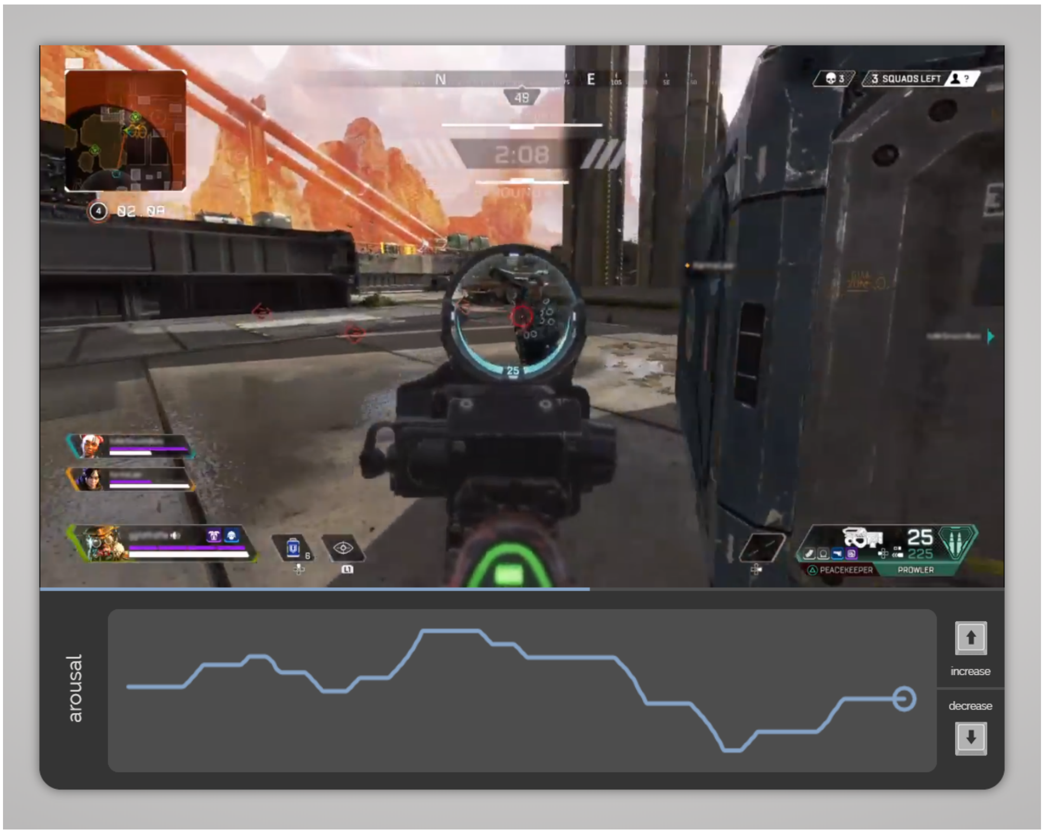


Figure 3.2: RankTrace interface in the PAGAN platform. *Apex Legends* (Electronic Arts, 2019) gameplay footage. No copyright infringement intended.

review).

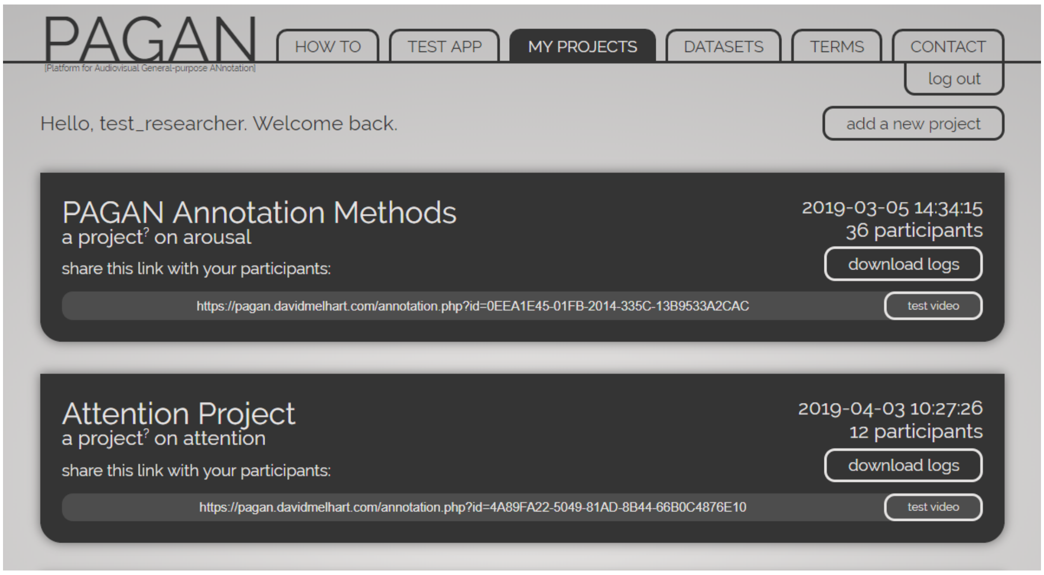
3.2 PAGAN Interface

This section describes the *Platform for Audiovisual General-purpose ANnotation* (PAGAN) framework, its user interface and general usage. The user interface of PAGAN consists of two separate sections. One is a web interface for researchers to prepare the annotation task (Section 3.2.1) and the other is an interface for annotation by end-users (Section 3.2.2). Section 3.2.3 details the three annotation methods incorporated currently in PAGAN.

3.2.1 Administration Interface

Researchers access and create their projects through a dedicated page. Each user has a secure login with a username and a password. After login, the researcher accesses their *project summaries* (Figure 3.3). Here, they can create new projects, view the progress of their ongoing studies, and access their corresponding annotation logs. Each project has a corresponding link, which is meant to be shared with potential participants. The annotation application can also be run in *test mode* from here, in which case the annotation logs are not saved.

The project creation screen can be seen in Figure 3.4. Projects are highly customisable to accommodate different research needs. The project title identifies the study on the



Hello, test_researcher. Welcome back.

PAGAN Annotation Methods
a project? on arousal
share this link with your participants:
https://pagan.davidmelhart.com/annotation.php?id=0EEA1E45-01FB-2014-335C-13B9533A2CAC
2019-03-05 14:34:15
36 participants

Attention Project
a project? on attention
share this link with your participants:
https://pagan.davidmelhart.com/annotation.php?id=4A89FA22-5049-81AD-8B44-66B0C4876E10
2019-04-03 10:27:26
12 participants

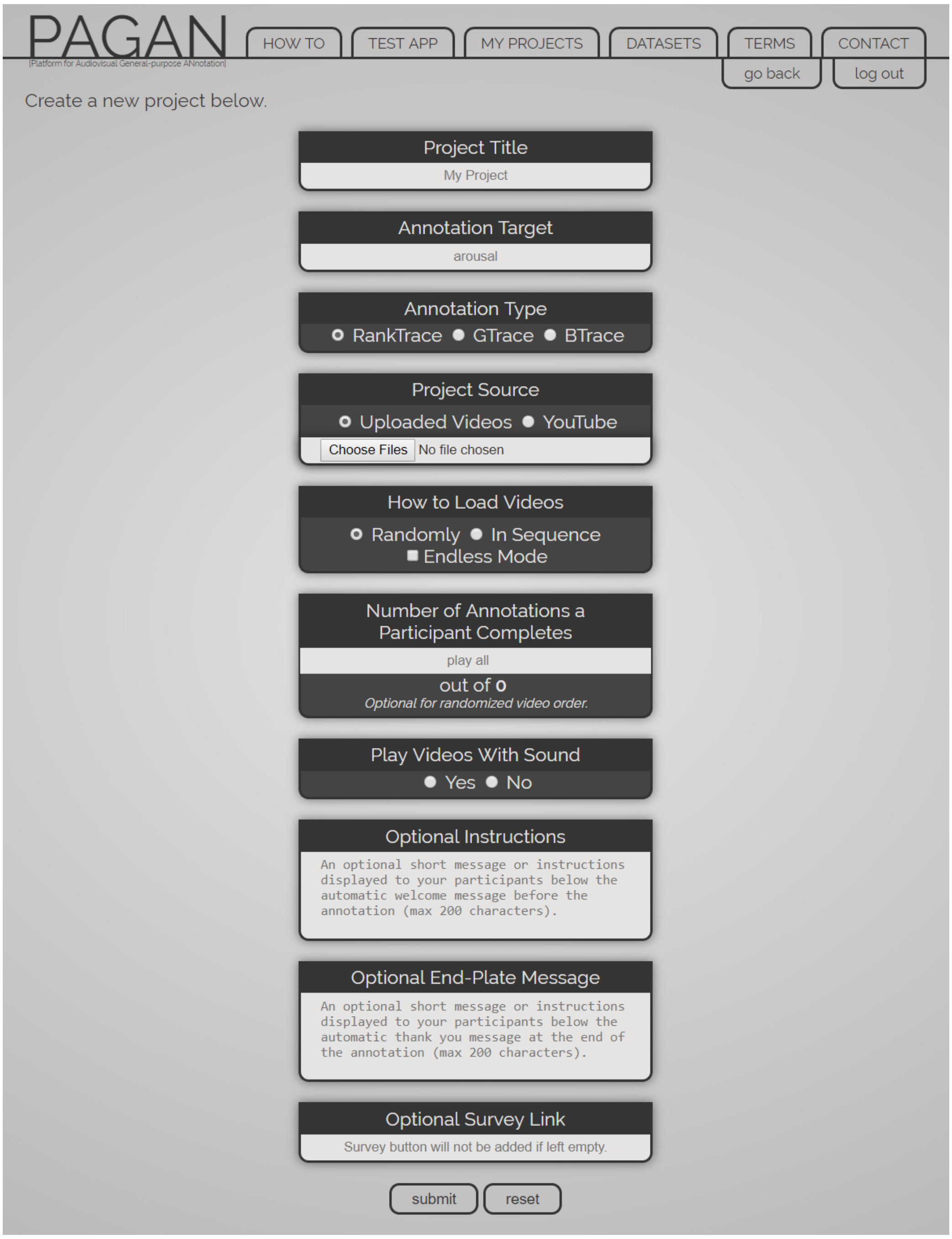
Figure 3.3: Project summaries on the administration interface.

project summary page and displays it to the participants as part of the welcome message. The *annotation target* is the label for the *y* axis of the annotator (see Figure 3.2). The project can be sourced from one or more uploaded videos or YouTube[3] links. The videos can be loaded either randomly or in sequence. If *endless mode* is selected, PAGAN rotates the videos indefinitely, allowing a participant to complete all tasks multiple times. In the case of a randomised video order, there is an option to limit the number of videos a participant has to annotate. The videos can be played with or without sound; if videos are played with sound, PAGAN reminds participants to turn on their speakers or headphones. The researcher can optionally add information or instructions viewed before and after the annotation tasks to help integrate the platform into the larger research project. Finally, a survey link can be included and displayed to the participant at the end of the annotation session.

3.2.2 Annotator Interface

The annotator application is a separate interface from the researcher site and meant to be used by the study participants. The interface is designed to display only the necessary information, thus eliminating potential distractions. Upon navigating to the project link (see Section 3.2.1), the participant is greeted by a welcome message which concisely explains the annotation procedure and provides some information about the annotation target (Figure 3.5). After the video is loaded, the participant can start the annotation process (Figure 3.2) at their leisure. While the original design of PAGAN eliminated the use of a computer mouse in favour of the more readily-available keyboard (Melhart et al., 2019b), an updated version of the platform uses a mouse scroll-wheel for the unbounded continuous annotation method. This change was made to recreate the more intuitive feeling of specialised annotation interfaces (Lopes et al., 2017b). In other tasks, the annotation is performed with the *up* and *down* keys on the keyboard, and the session can be paused by pressing *space*. The system only logs a session as "completed" if at least 25% is seen

[3] `https://www.youtube.com/`



Create a new project below.

Project Title
My Project

Annotation Target
arousal

Annotation Type
◉ RankTrace ○ GTrace ○ BTrace

Project Source
◉ Uploaded Videos ○ YouTube
Choose Files No file chosen

How to Load Videos
◉ Randomly ○ In Sequence
□ Endless Mode

Number of Annotations a Participant Completes
play all
out of 0
Optional for randomized video order.

Play Videos With Sound
○ Yes ○ No

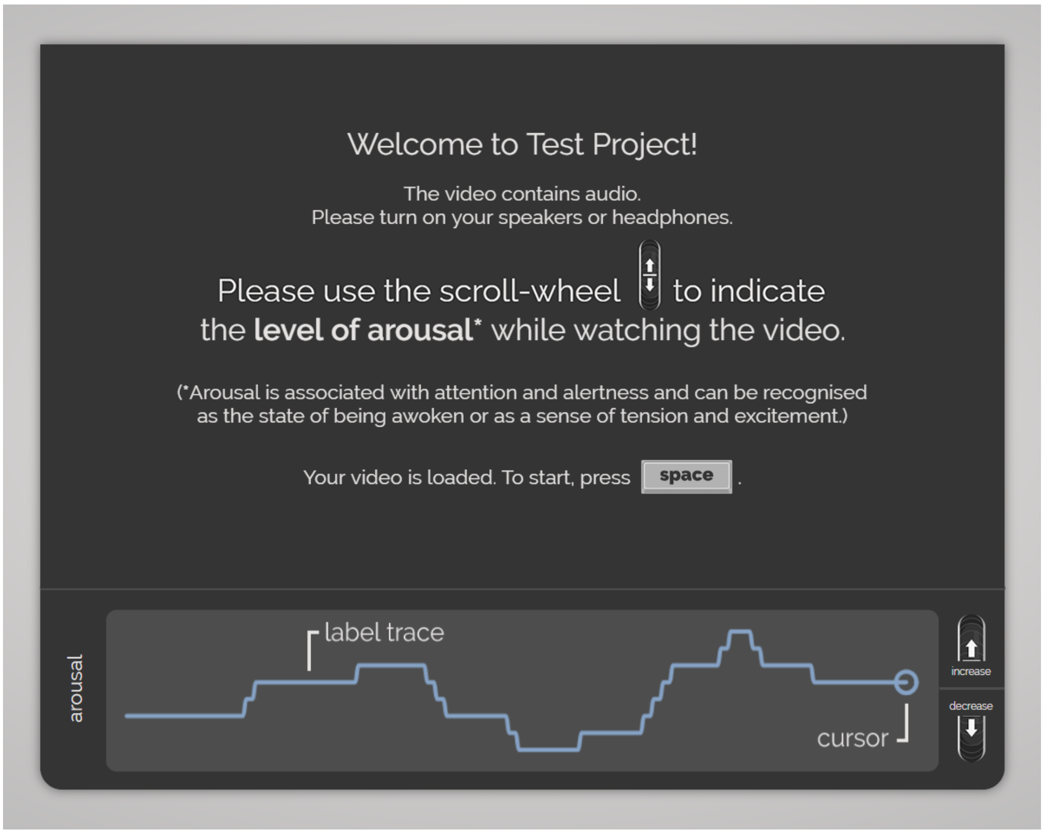
Optional Instructions
An optional short message or instructions displayed to your participants below the automatic welcome message before the annotation (max 200 characters).

Optional End-Plate Message
An optional short message or instructions displayed to your participants below the automatic thank you message at the end of the annotation (max 200 characters).

Optional Survey Link
Survey button will not be added if left empty.

Figure 3.4: Project creation screen on the researcher interface.

and pauses if the browser tab is out of focus (i.e. if the participant leaves the annotation interface open but switches to a different tab or window) to minimise the number of sessions with insufficient annotation.



Welcome to Test Project!

The video contains audio.
Please turn on your speakers or headphones.

Please use the scroll-wheel to indicate the **level of arousal*** while watching the video.

(*Arousal is associated with attention and alertness and can be recognised as the state of being awoken or as a sense of tension and excitement.)

Your video is loaded. To start, press space .

Figure 3.5: Welcome message displayed to participants before annotation starts.

3.2.3 Annotation Methods

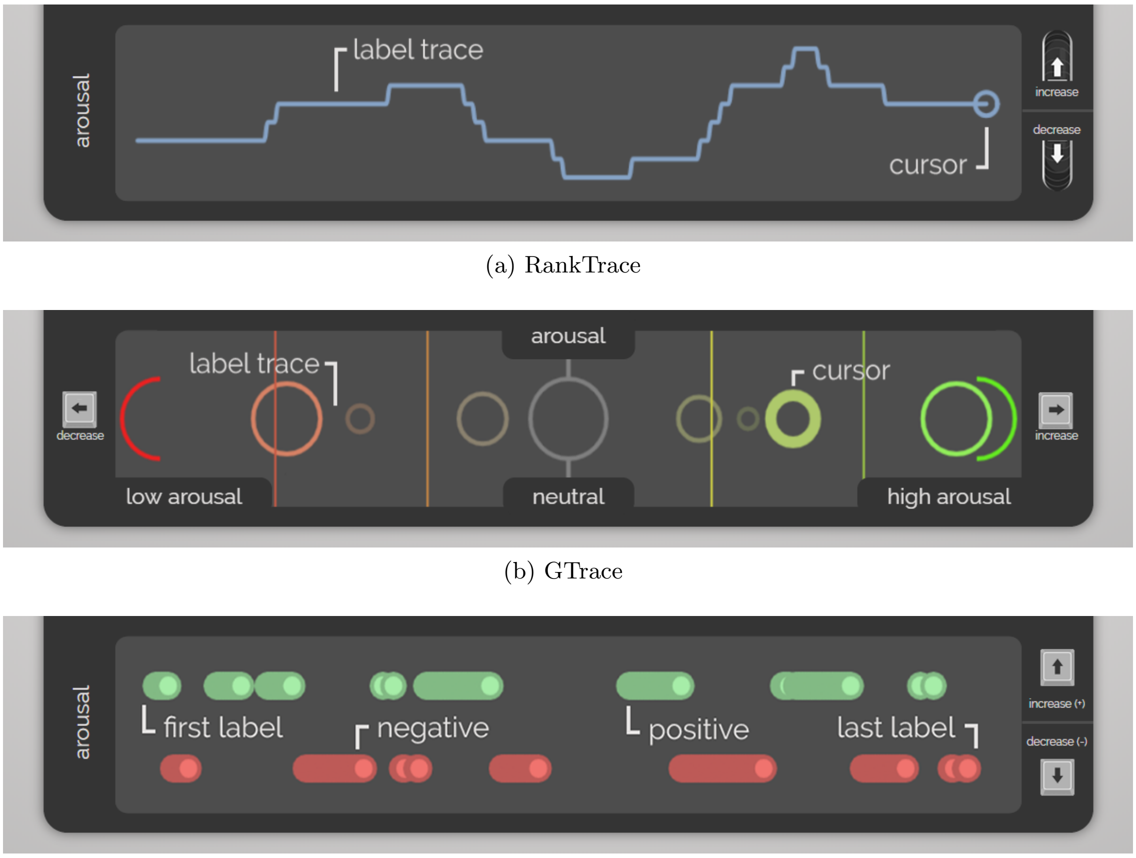
This section presents the annotation techniques included in the PAGAN framework: RankTrace, GTrace, and BTrace.

3.2.4 RankTrace

The implementation of RankTrace closely follows the original by Lopes et al. (2017b) (see PAGAN implementation on Figure 3.6a). The only major distinction to their version is using a mouse scroll wheel to move the annotator cursor instead of the specialised interface used with RankTrace. This change was made to approximate the feeling of the original version of RankTrace—which uses a wheel interface to control the magnitude of change—without any specialised equipment. As the annotation trace displays the entire history, the participant has sufficient visual feedback, which acts as a reference (anchoring) point (Yannakakis et al., 2018) for the subjective evaluation of the experience.

3.2.5 GTrace

Similarly to how the tool is used in Baveye et al. (2015a), the user interface is moved under the video; vertical lines are added as an allusion to a traditional 7-item scale to provide a visual aid for the absolute evaluation of the trace (see the PAGAN implementation on Figure 3.6b). The main distinction from the original GTrace variant is that the cursor's movement is accelerated when a key is held down. This change was necessary as the



(a) RankTrace

(b) GTrace

(c) BTrace

Figure 3.6: Interfaces for the annotation methods included in this study.

original implementation used a mouse cursor or joystick, allowing for higher speed while retaining precision. When the participant stops the cursor, it leaves a mark that slowly fades, providing limited memory of previous positions, to which the participant can compare new labels. The limited memory from the fading mark differs from BTrace and RankTrace that display the entire history of the session.

3.2.6 BTrace

Binary Trace (BTrace) is a new annotation tool introduced in PAGAN (Melhart et al., 2019b) which is primarily based on AffectRank (Yannakakis and Martinez, 2015). BTrace is designed as a simple alternative to relative annotation in a discrete manner, using two nominal categories: +1 as the increase (or positive change) and −1 as the decrease (or negative change). In that regard, it could be viewed as a one-dimensional version of AffectRank. The design of the tool, however, is based on the benefits reference points have on the reliability of the obtained annotation labels (Yannakakis et al., 2018; Lopes et al., 2017b) and thus, it displays the entire history of the annotation session as red and green blobs (see Figure 3.6c).

Table 3.1: Summary of participants' choice of most and least interesting video and most and least intuitive annotation tool

Most / Least Interesting Video		
Apex	GoT	SEMAINE
11 / 6	15 / 1	3 / 22

Most / Least Intuitive Tool		
RankTrace	GTrace	BTrace
16 / 4	8 / 8	5 / 17

Table 3.2: Number of annotation traces for each video and annotation type, the average number of samples are shown in brackets

Annotation Type	**Apex**	**GoT**	**SEMAINE**
RankTrace	11 (229)	10 (514)	10 (425)
GTrace	11 (429)	12 (358)	8 (133)
BTrace	9 (302)	8 (120)	14 (85)

3.3 Usability Study

To identify the best annotation method for subsequent data collection (see Chapter 4), a small-scale study was conducted with 36 participants. This study focuses on the perceived arousal level of different videos with emotional content and examines the effectiveness of relative annotation methods (RankTrace and Btrace) compared to absolute affect labelling (GTrace). For the sake of comprehensiveness, the study incorporates three different types of elicitors: videogame footage, movie trailer, and a recorded conversation.

3.3.1 Collected Data

The collected data consists of 108 annotated videos from 36 participants. Participants were found through the social and academic network of the researchers, while subsequent parties were added through snowball sampling by participants sharing the project link. The average age of the participants is 29 years old and 69% identified as male, 24% identified as female, one subject identified as queer, and one did not want to identify themselves. The majority of the participants were avid gamers, with 59% playing more than once a week. Each participant was asked to annotate three videos with different but emotionally evocative content: (a) recorded gameplay from *Apex Legends* (Electronic Arts, 2019) (Apex), a popular Battle Royale-style game; (b) the Season 8 trailer of the TV series *Game of Thrones* (HBO, 2019) (GoT); (c) a conversation between a human participant and "Spike", the angry virtual agent in the *SEMAINE* database (McKeown et al., 2012). All videos are approximately 2 minutes long. Each video was assigned a random annotation type, and the order of videos was also randomised.

Participants were asked to name the most and least interesting of the three videos and the most and least intuitive of the three annotation tools, effectively ranking them. The results of their preferences are summarised in Table 3.1. The GoT trailer was the most popular (only one participant rated it as the least interesting), while the video from the SEMAINE database was by far the least liked (it collected 81% of "least interesting" votes).

In terms of usability, participants ranked RankTrace the most intuitive (as it received 55% of "most intuitive" votes), GTrace second, and BTrace the least intuitive.

3.3.2 Methodology

To measure the reliability of the different annotation techniques over the different videos, we observe the inter-rater agreement between participants. Inspired by Yannakakis and Martinez (Yannakakis and Martinez, 2015), the inter-rater agreement is measured with the *Krippendorff's α* coefficient (Krippendorff, 2004), which is a robust metric of the degree of agreement corrected for a chance between any number of observers and any type of data. Krippendorff's $\alpha = 1 - D_o/D_e$, where D_e denotes the expected and D_o the observed disagreements between annotations. Krippendorff's α is adjusted to the level of measurement of the observations through the weighing of the expected and observed coincidences (see Krippendorff (2011) for a complete explanation). This robustness allows for a fair comparison between different annotation methods. Krippendorff's α has an upper bound of 1, indicating absolute agreement, while 0 signifies no agreement or pure chance. At Krippendorff's $\alpha < 0$, disagreements between annotators are systematic and go beyond chance-based levels.

To compare discrete and continuous annotation and smooth out some of the surface differences between individual traces, we compartmentalise the signals into equal length time-windows. This method of preprocessing is often used in affective computing to preprocess time-continuous signals (Yannakakis and Martinez, 2015; Camilleri et al., 2017; Lopes et al., 2017b). We clean the dataset of traces which either had extremely few samples from annotation (less than 3) or where viewing time was less than a minute. This cleanup process removed 15% of traces, and the final datasets are comprised of 92 traces. Table 3.2 shows the number of traces and samples in each dataset and annotation method. In this study, 3-second time-windows are considered without any overlap. Potentially the 3-second processing provides approximately 40 windows per participant. As some participants did not complete the entire annotation task, this number can vary. However, to maximise the sample sizes, we decided to keep these traces as Krippendorff's α can be applied to data with missing observations.

As BTrace already encodes perceived change, similarly to AffectRank (Yannakakis and Martinez, 2015), the values of time-windows are computed as the *sum of annotation values* (ΣA) within each window, adding values in case of increase and subtracting them in case of decrease. For RankTrace and Gtrace, both an absolute and a relative metric are considered (Camilleri et al., 2017): the mean value (μA) and average gradient (∇A) of time-windows based on the min-max normalised traces. The mean value is considered an absolute metric because it denotes the general level of the participant's response in a given time-window. In contrast, the average gradient of a time-window considers the amount and direction of the change that happened, as it is computed from the differences of adjacent datapoints of the trace (Camilleri et al., 2017; Lopes et al., 2017b). The calculation of Krippendorff's α is adjusted to the observed metric. When the annotation trace is processed into a relative metric (ΣA, ∇A), the annotation values are compared as ordinal variables. When the annotation trace is processed into an absolute metric (μA), the annotation values are compared as interval variables.

Table 3.3: Krippendorff's α across annotation traces processed as 3-second time-windows

Annotation		Video		
Tool	**Processing**	**Apex**	**GoT**	**SEMAINE**
RankTrace	∇A	0.2025	0.1760	-0.0043
	μA	0.1542	-0.0227	0.0147
GTrace	∇A	0.1517	0.1224	-0.0347
	μA	0.1857	0.0549	0.0926
BTrace	ΣA	0.3193	0.0973	0.0249

3.3.3 Results

Table 3.3 shows the calculated inter-rater agreement. For comparing between RankTrace and GTrace, the highest α value between ∇A and μA is used. The top α values for RankTrace are 0.20 for Apex and 0.18 for GoT, which are higher than the highest α values for GTrace (0.19 and 0.12, respectively). For both GoT and Apex videos, the highest α values are found with ∇A in three of the four instances examined (except for annotations with GTrace on the Apex dataset), which is further evidence that processing time-windows of GTrace ratings through a relative measure yields more consistent results. Interestingly, both GTrace and RankTrace have a higher α value with μA for the SEMAINE video (with GTrace having superior performance), although generally these values are very low and any inter-rater agreement could be chance-based. The general findings from these comparisons are in line with a growing body of research promoting the relative collection and processing of affective annotation traces (Yannakakis and Martínez, 2015; Camilleri et al., 2017; Yannakakis et al., 2018).

Based on Table 3.3, it seems that BTrace achieves the highest inter-rater agreement on the Apex dataset while showing lower reliability on GoT and SEMAINE videos. As the compartmentalised binary labels denote the rough amount of perceived change in a time-window (but not its magnitude), the possibility of relatively high inter-rater agreement is not surprising. However, results on the GoT and SEMAINE videos show the unreliability of this method. A possible reason for the high variance in the inter-rater agreement is the low face validity of the method. BTrace collected 59% of the "least intuitive" votes among the three annotation methods. Therefore, despite its potential robustness in some instances, BTrace has shown to be the least reliable and intuitive to use.

An unexpected finding of this analysis is the overall low inter-rater agreement of all methods on the chosen SEMAINE video (ranked as the least interesting by the participants). A plausible explanation of these results is a connection between the context and intensity of the affective content and the reliability of the annotation traces. While games and trailers are designed to elicit arousal, the slow pace of SEMAINE videos can be unappealing by comparison. The differences in inter-rater agreements between the Apex and GoT datasets also point towards the role of context in emotion elicitation. While the GoT video is authored to elicit high arousal, the Apex footage presents a more organic scenario with relative calm periods and high-octane action. Especially for frequent videogame players, who have personal experiences with the dynamics of shooter games, this video is easier to interpret, and the affective high-points are easier to recognise. This is also supported by a recent study by Jaiswal et al. (2019), who observed an effect between the context of the annotation task and the quality of labels.

3.3.4 Conclusions

The usability study included three videos indicative of different sources of arousal: a game, a TV series trailer, and a dialogue with a virtual agent. The analysis reveals low inter-rater agreement on the SEMAINE database video, which raises the question of whether more engaging forms of emotion elicitation such as games would offer more reliable benchmarks for affective computing research. Results demonstrated the reliability of the supported annotation techniques and showed the strength of relative annotation processing. While BTrace can produce high reliability, RankTrace is the most intuitive tool to use for participants. As BTrace is also limited in capturing more minor deviations in the user experience, subsequent studies in this thesis will use RankTrace to capture annotations.

3.4 Summary

This chapter described a pipeline for the first-person annotation of gameplay. The presented techniques are inspired by stimulated recall methods and extend the PAGAN annotation framework into a complete data collection pipeline for games user research. The latter half of the chapter detailed the PAGAN framework for multimedia annotation. This platform was designed to enable the easy crowdsourcing of annotation tasks generally used in affective computing. While the tool includes several popular one-dimensional and time-continuous annotation techniques, through a small-scale usability study, it has been shown that an unbounded continuous trace (RankTrace) has the most utility to annotate arousal in videogames.

The next chapter presents the dataset collected using the PAGAN framework, and the first-person affect annotation pipeline introduced early in this thesis. Later, the next chapter focuses on the captured telemetry and annotation traces in a preliminary analysis, including a qualitative and statistical overview.

Chapter 4

The AGAIN Dataset

While much advancement has been made in general game playing (Perez-Liebana et al., 2016b) with the advent of deep learning (Torrado et al., 2018), the use of general algorithms to predict affective outcomes received much less attention so far (Togelius and Yannakakis, 2016). One significant barrier of entry to the field is the lack of large, reliable, and open datasets. The handful of studies focusing explicitly on affect prediction have been using ad-hoc datasets with relatively small sample sizes (Shaker et al., 2015; Shaker and Abou-Zleikha, 2016; Camilleri et al., 2017). The lack of a publicly available database for general affect modelling means that researchers have to allocate resources to building their own corpus often using limited resources. While early studies showed some success in transferring models between dissimilar games, their results are hard to compare and build upon in lieu of a shared baseline. If general player affect modelling is to develop, it needs large open datasets that are publicly available.

The creation of the *Affect Game AnnotatIoN* (AGAIN) dataset was motivated by the lack of corpora for the study of general properties of affect across tasks and participants. AGAIN contains data from over 120 participants who played and annotated over $1,000$ gameplay sessions. It is accessible online[1] and features data collected from nine games spanning across three dissimilar genres, which were developed specifically for the purposes of the dataset (see Figure 4.1). As shown in Table 4.1, along with game telemetry and self-annotated *arousal* labels, the dataset also features a video database of unique gameplay sessions with over 37 hours of in-game footage. The diverse nature of the AGAIN affect elicitors (games) provides a testbed for general affect detection in games Togelius and Yannakakis (2016); Camilleri et al. (2017) and broadens the horizons for further research on general-purpose AI representations Makantasis et al. (2019); Mnih et al. (2015) and artificial general intelligence.

The design and creation of AGAIN was guided by the following factors: a) *accessibility*, which is achieved through an online crowdsourcing framework; b) *scalability*: AGAIN is utilising the PAGAN online annotation framework (see Chapter 3) and, hence, one can quickly populate the AGAIN database with more participants and annotators; c) *extensibility*: more affect dimensions and categories can be considered and integrated to the existing dataset through the customisable PAGAN annotation tool, and; d) *generality*: any other online game or interactive session can be easily integrated to the experimental protocol of AGAIN. While at the time of writing, the dataset hosts nine games annotated for arousal AGAIN is designed with all aforementioned factors in mind so that it can host data from

[1] `http://again.institutedigitalgames.com/`

Table 4.1: Core Properties of the AGAIN Dataset

Properties	Raw dataset	Clean dataset
Number of Participants	124	122
Number of Gameplay Videos	1116	995
Number of Game-telemetry Logs	1116	995
Video database size	37+ hours	33+ hours
Number of Elicitors	9 games (3 genres)	
Gameplay/Video duration	2 min	
Annotation Perspective	First-person	
Annotation Type	Continuous unbounded	
Affective Labels	Arousal	

more games and user modalities, considering alternative affective labels.

The AGAIN dataset is unique in a number of ways. First, it is the largest and most diverse publicly available affective dataset based on games as interactive elicitors. Given the breadth of elicitors offered, the dataset can be used for testing specific affect models on one particular task (i.e. a particular game) all the way to general models of affect across tasks (game genres and games in general). Second, the dataset is annotated with the core affective dimension of arousal, linking dominant annotation practices in affective computing with player modelling and game user research. Finally, it employs a novel annotation framework Lopes et al. (2017b) which captures subjective annotations in a continuous and unbounded manner that can be further processed as labels for regression, classification or ordinal learning affect modelling tasks Yannakakis et al. (2018, 2017). Section 4.1 discusses the motivation and different considerations that were taken into account while building the dataset. Section 4.2 introduces the games used in the dataset. Section 4.3 describes the data collection process, the collected data, and the cleaning procedure applied to it. First, Section 4.4.1 explores general trends in the cleaned AGAIN dataset. This section also deals with the distribution of the output feature, *arousal*, and looks at the patterns in the input features, focusing mainly on general features introduced in Section 4.3.4. Section 4.4.2 presents additional pre-processing steps which are necessary for subsequent machine learning tasks. Finally, Section 4.4.3 presents a correlation analysis of the pre-processed features. This section provides insights on the linear relationships within the data, which can also help explain some of the more surprising results later. Section 4.5 provides a summary of the Chapter.

4.1 From General Models of Affect to Games and Arousal

Traditional modelling approaches—both supervised and unsupervised (see Chapter 2)—rely on large amounts of data to build reliable models. Apart from the raw data need, predictive modelling also has to solve the issue of data labelling. However, with an ever-increasing need for larger and larger datasets, supervised player modelling finds itself at a disadvantage as collecting reliable data becomes more and more unfeasible. Beyond the feasibility, the utility of game-based models is also limited. A clear drawback of this approach is that the resulting models are only applicable to games that are already published or in an *early-access* phase, where introducing large changes to the game is impossible. Not surprising then that most of the industry application of player modelling focuses on market research (Viljanen et al., 2018). In contrast, general models can make valuable real-time predictions of the player experience (Makantasis et al., 2019; Melhart et al., 2020a), behaviour (Bakkes et al., 2012), motivation (Melhart, 2018), and affective state (Lopes et al., 2017a; Camilleri et al., 2017) as well, which could inform processes varying from design decisions to monetisation. The value of player modelling is clear; however, the resource need of building these models so far prevented their large-scale industrial use. As the AAA industry shifts towards the"games as a service" model, which introduces large content packs, often restructuring games in a major way, the need for adaptable player models, which rely on minimal data, is greater than ever. Consequently, general player modelling has been making advancements, especially in the field of general-game playing (Togelius and Yannakakis, 2016; Torrado et al., 2018). While these models have their uses, they tell less about how players interact with the game. To answer this issue, general affect modelling (Togelius and Yannakakis, 2016) aims to create pre-trained models, which can be applied to unseen games. If successful, such models can reduce the data need of new projects. While research has been started in this field, studies in the literature are still rather sparse.

Beyond industrial applications, general game-playing is also a major step towards the understanding of a universal human experience, and subsequently, general artificial intelligence (Togelius and Yannakakis, 2016). Affective player modelling can also provide value to other fields of game research occupied with game-playing and content generation. Through an *affective loop*, which connects emotion expression, elicitation, detection, prediction, and reaction, a game system can facilitate different emotional states or mimic emotional human behaviour (Yannakakis and Togelius, 2018). Such a system can use emotion prediction as input for game-playing agents and procedural content generation systems, and other types of facet-orchestration (Liapis et al., 2018). While there are a few studies (Yannakakis and Hallam, 2009; Shaker et al., 2010; Melhart et al., 2020c) and commercial games (e.g. *Façade* (Procedural Arts, 2005) and *Nevermind* (Flying Mollusk, 2016)) demonstrating the feasibility of a fully realised affective loop, research in this direction is still in its infancy. One way to break new ice in the field is to apply general affect models, which can jump-start projects with reliable "off the shelf" affect predictions for other generating and orchestrating systems.

As discussed in Chapter 2, *arousal* is one of the main building blocks of human emotion. In the affective literature, it is one of the more consistent components of dimensional emotion models, and in the domain of videogames, arousal carries rich information about how the players appraise the game. The intensity of the game described by arousal facilities the play dynamics and the perceived challenge of the game (Klarkowski et al., 2016) which affects cognitive and emotional engagement (Abbasi et al., 2019) and can lead to a number of psychological outcomes including tension (Lopes et al., 2017a), frustration (Melhart, 2018), fun (Clerico et al., 2016), and flow (Seger and Potts, 2012), as well as positive post-game outcomes, such as increased creativity (Yeh, 2015) and working memory (Gabana et al., 2017) performance. Due to this special relevance to videogames, the collected data for the AGAIN dataset was labelled using arousal. Focusing on one affect dimension reduces the cognitive load of the annotation task (Melhart et al., 2019b), which in turn increases the reliability of the data; however, it limits the expressive range of affect annotation in the dataset. Moreover, the focus on arousal assists the research community to build, extend upon and advance studies that already have benchmarked the study of arousal in games (Lopes et al., 2017b; Camilleri et al., 2017; Makantasis et al., 2019). Thus, it stands to reason that a systematic overview of general affect modelling in games starts from arousal as an information-rich background.

Collecting a large dataset is not a trivial task. Developing testbed games, designing and deploying the necessary software infrastructure, and recruiting participants all take a considerable amount of time and resources. The collection process is then constrained by the available time and resources and the experimental design of the collection process. Because long annotation sessions increase participants' cognitive load and stimulated recall techniques rely on a rapid recall of recent episodic memory, the annotation process must be kept relatively short. Therefore, there is a trade-off between how close we can model real-life experiences, how much data we can collect and how intrusive the process can be. Unfortunately, a AAA game experience cannot be emulated with the given constraints as those games are far more ambitious in size and fidelity. This is especially true for games in the adventure, role-playing, and strategy genres that require a considerable time commitment from the players. Nevertheless, more casual experiences can be approximated in simpler testbeds as well.

The games included in the dataset are approximations of popular genres of both the PC and mobile markets. Nine games from three different genres have been designed and developed exclusively for the AGAIN dataset. The genres chosen were *Racing*, *Shooter*, and *Platformer*. It speaks to the popularity of these genres that in the top seller chart of *Steam*[2] one can often see different iterations on the *Shooter* and *Platformer* genres. At the time of writing, the third on the top seller list is a platformer, *Ori and the Will of the Wisps* (Moon Studios, 2020) and fourth is a shooter, *Counter-Strike: Global Offensive - Operation Broken Fang* (Valve & Hidden Path Entertainment, 2020). While the mobile market is dominated by puzzle, simulation, and idle games, in the top downloads and top-grossing charts of *Google Play*[3], one can find multiple examples of casual shooters such as *Brawl Stars* (Supercell, 2017), platformers such as *Super Mario Run* (Nintendo, 2016) and racing games such as *CSR Racing 2* (Zynga, 2013). The main considerations when developing the games in the dataset, was to create examples which are representative of their genre, easy to play and understand with a rudimentary level of game literacy (Buckingham and Burn, 2007), and aesthetically pleasing to get a close approximation of what players experience in the wild. Designing and developing the games from the ground-up instead of relying on available testbed games—such as Mazeball (Yannakakis et al., 2010) or Sonancia (Lopes et al., 2017a)—also had the added benefit of control over the data logging procedure, resulting in a more coherent and consistent database without the need of heavy pre-processing, which can introduce biases into the dataset.

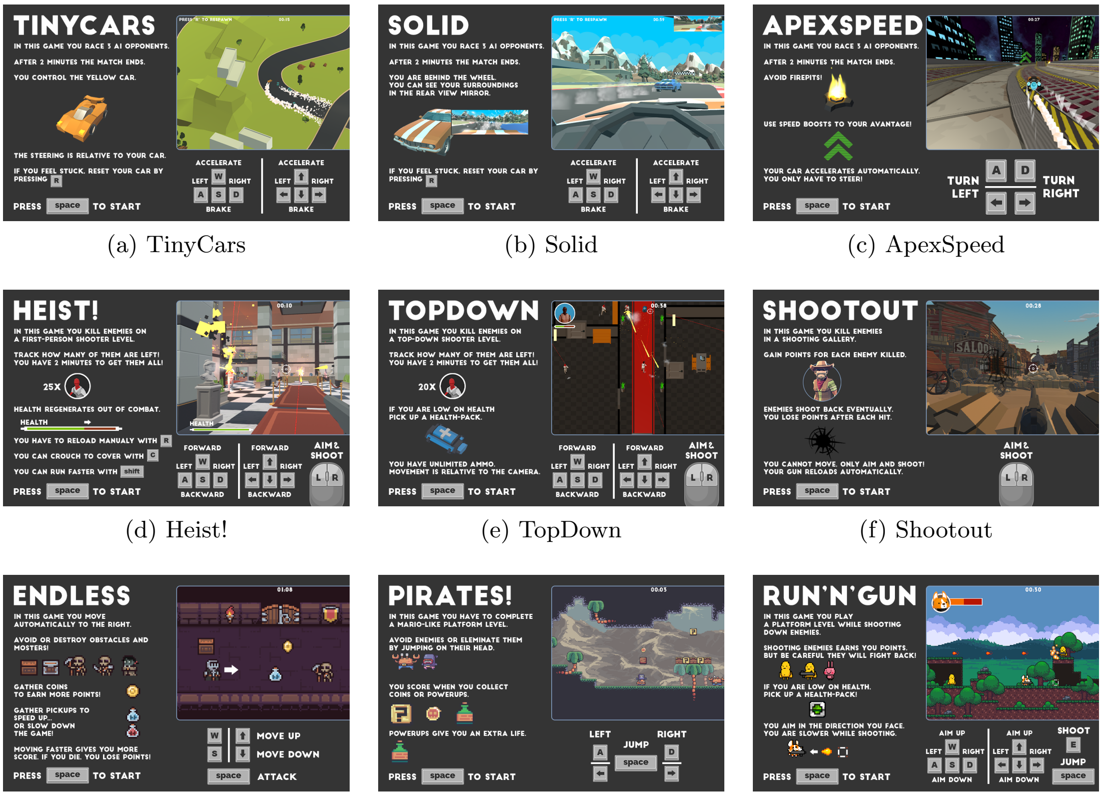
4.2 Game Descriptions

Games were designed in Unity 3D[4] and built for the WebGL platform. The selected genres represent a good cross-section of popular games (Yannakakis and Togelius, 2018; Sevin and DeCamp, 2020; Vargas-Iglesias, 2020), which rely on fast-paced gameplay and hand-eye coordination. While this focus omits other popular genres, such as role-playing or strategy games, the included genres require substantially less time to learn and play successfully, making them unfeasible for the proposed data collection procedure. The collected dataset was published independently under the name, AGAIN (Affect Game Annotation) Database

[2] `http://store.steampowered.com/`

[3] `https://play.google.com/`

[4] `https://unity.com/`



TINYCARS

IN THIS GAME YOU RACE 3 AI OPPONENTS.

AFTER 2 MINUTES THE MATCH ENDS.

YOU CONTROL THE YELLOW CAR.

THE STEERING IS RELATIVE TO YOUR CAR.

IF YOU FEEL STUCK, RESET YOUR CAR BY PRESSING R

PRESS space TO START

(a) TinyCars

SOLID

IN THIS GAME YOU RACE 3 AI OPPONENTS.

AFTER 2 MINUTES THE MATCH ENDS.

YOU ARE BEHIND THE WHEEL.
YOU CAN SEE YOUR SURROUNDINGS
IN THE REAR VIEW MIRROR.

IF YOU FEEL STUCK, RESET YOUR CAR BY PRESSING R

PRESS space TO START

(b) Solid

APEXSPEED

IN THIS GAME YOU RACE 3 AI OPPONENTS.

AFTER 2 MINUTES THE MATCH ENDS.

AVOID FIREPITS!

USE SPEED BOOSTS TO YOUR AVANTAGE!

YOUR CAR ACCELERATES AUTOMATICALLY.
YOU ONLY HAVE TO STEER!

PRESS space TO START

(c) ApexSpeed

HEIST!

IN THIS GAME YOU KILL ENEMIES ON A FIRST-PERSON SHOOTER LEVEL.

TRACK HOW MANY OF THEM ARE LEFT!
YOU HAVE 2 MINUTES TO GET THEM ALL!

25X

HEALTH REGENERATES OUT OF COMBAT.

YOU HAVE TO RELOAD MANUALY WITH R

YOU CAN CROUCH TO COVER WITH C

YOU CAN RUN FASTER WITH shift

PRESS space TO START

(d) Heist!

TOPDOWN

IN THIS GAME YOU KILL ENEMIES ON A TOP-DOWN SHOOTER LEVEL.

TRACK HOW MANY OF THEM ARE LEFT!
YOU HAVE 2 MINUTES TO GET THEM ALL!

20X

IF YOU ARE LOW ON HEALTH PICK UP A HEALTH-PACK.

YOU HAVE UNLIMITED AMMO.
MOVEMENT IS RELATIVE TO THE CAMERA.

PRESS space TO START

(e) TopDown

SHOOTOUT

IN THIS GAME YOU KILL ENEMIES IN A SHOOTING GALLERY.

GAIN POINTS FOR EACH ENEMY KILLED.

ENEMIES SHOOT BACK EVENTUALLY.
YOU LOSE POINTS AFTER EACH HIT.

YOU CANNOT MOVE, ONLY AIM AND SHOOT!
YOUR GUN RELOADS AUTOMATICALLY.

PRESS space TO START

(f) Shootout

ENDLESS

IN THIS GAME YOU MOVE AUTOMATICALLY TO THE RIGHT.

AVOID OR DESTROY OBSTACLES AND MOSTERS!

GATHER COINS TO EARN MORE POINTS!

GATHER PICKUPS TO SPEED UP...
OR SLOW DOWN THE GAME!

MOVING FASTER GIVES YOU MORE SCORE. IF YOU DIE, YOU LOSE POINTS!

PRESS space TO START

(g) Endless

PIRATES!

IN THIS GAME YOU HAVE TO COMPLETE A MARIO-LIKE PLATFORM LEVEL.

AVOID ENEMIES OR ELEMINATE THEM BY JUMPING ON THEIR HEAD.

YOU SCORE WHEN YOU COLLECT COINS OR POWERUPS.

POWERUPS GIVE YOU AN EXTRA LIFE.

PRESS space TO START

(h) Pirates!

RUN'N'GUN

IN THIS GAME YOU PLAY A PLATFORM LEVEL WHILE SHOOTING DOWN ENEMIES.

SHOOTING ENEMIES EARNS YOU POINTS.
BUT BE CAREFUL THEY WILL FIGHT BACK!

IF YOU ARE LOW ON HEALTH, PICK UP A HEALTH-PACK!

YOU AIM IN THE DIRECTION YOU FACE.
YOU ARE SLOWER WHILE SHOOTING.

PRESS space TO START

(i) Run'N'Gun!

Figure 4.1: Start screens of the nine games included in the AGAIN dataset, showing the game's rules and players' controls.

(Melhart et al., 2021)[5].

4.2.1 Racing Games

The *racing* genre includes car-racing simulators or close approximations of the same experience. Games in this genre feature fast-paced gameplay and rely on dexterity. While driving along a racetrack, players have fewer interactions with artificial opponents than in other games included in the dataset, but some racing games still allow players to crash into and push opponents off the track to force them into a worse position. In all included games, the races are played in a closed loop. Players start from the last position and have to fight their way to the top during the race. While there are no combat mechanics, opponents and the environment can pose extra challenges to the player. If the player is stuck, they can reset their position to the last checkpoint by pressing '*R*'.

- **TinyCars** is an isometric arcade-style racing game (see Fig. 4.1a). The player's view is fixed in a 45 deg top-down angle. The controls are relative to the player's car. The main challenge of the game is keeping the car on track, as going off-track comes with a substantial speed penalty. While there are no large obstacles on the level, the track does feature an overpass and a small jump-ramp. The game was inspired by the

[5] http://again.institutedigitalgames.com/

retro-racer game, *Super Cars II* (Magnetic Fields, 1991).

- **Solid** is a rally game (see Fig. 4.1b), where the player's view is fixed inside the car, with the steering wheel and dashboard partially obstructing the player's view. To aid the player's situational awareness, the UI features a rear-view mirror. The main challenge of the game is posed by the opponents as they bump into the player and each other. While there are no hefty penalties for leaving the track, there is a large loop on the level. The player has to hit a high enough speed to clear the obstacle. The game was inspired by games in the *Colin McRae Rally* series (Codemasters, 1998-2019).

- **ApexSpeed** is a speed racer-type game with a third-person view. As the camera follows the car from an elevated angle, the player has improved situational awareness compared to a first-person view. The main obstacles of the game are the fire-pits. On collision, these obstacles reset the player to the last checkpoint. While opponents can still be pushed aside, there is no speed penalty for Collision, and the level is closed, making it impossible to go off-track. The game was inspired by games like *Wipeout* (Psygnosis, 1995)—or more recently *Redout* (34BigThings, 2016).

4.2.2 Shooter Games

The *shooter* genre includes games that are characterised by action and combat. The goal of these games is to eliminate opponents by shooting at them with projectile weapons. Shooter games rely heavily on hand-eye coordination and fast reflexes. In all presented shooter games, the player has to use keyboard keys to navigate (except Shootout, where they are stationary) and the mouse to aim. These are the only games in the dataset that require a mouse to play.

- **Heist!** is a first-person shooter game (see Fig. 4.1d). The game imitates modern shooter games. The player can sprint and crouch behind covers. Their health is automatically regenerated after a few seconds out of combat. The main challenge of the game is managing the combat and ammunition. The player's weapon is a semi-automatic pistol, which has to be reloaded manually by pressing '*R*'. If the player runs out of health, they are respawned at the beginning of the level. The game was inspired by shooter games such as *Call of Duty: Modern Warfare* (Infinity Ward & Sledgehammer Games, 2019).

- **TopDown** is a third-person isometric shooter (see Fig. 4.1e). The player's view is fixed at a 45 deg top-down angle. Instead of controlling the camera gaze, the player moves the reticle with their mouse. The player's weapon is an automatic rifle with unlimited ammunition. The main challenge of the game is dodging enemy bullets while engaging in combat. Health does not regenerate, but the player can pick up *health-packs* to replenish some of it. Similarly to Heist!, when the player runs out of health, they are respawned at the beginning of the level with full health. The game was inspired by modern iterations on the top-down shooter genre, like *Neon Chrome* (10tons Ltd. 2016).

- **Shootout** is an arcade shooter (see Fig. 4.1f). The player cannot move, only aim and shoot with their mouse. The main challenge of the game is hand-eye coordination, reflexes, and managing ammunition. The player's weapon is a revolver that cannot be reloaded manually. When the player runs out of ammunition, the gun is automatically reloaded (preventing shooting for 2 seconds). The game does not have a health system, but the player loses points whenever they are hit. Enemies appear on screen at an increasing rate and numbers, slowly overwhelming the player. The game was inspired by classic shooting gallery games like *Hogan's Alley* (Nintendo, 1984).

4.2.3 Platformer Games

The *platformer* genre focuses on traversal, often with light puzzles with gameplay that requires dexterity and precision. Platformers often feature adversaries (just as in shooter games); however, the goal in these games is just as often to avoid these enemies as to eliminate them. The platformer set of the database is the most diverse among all genres, with more diverse control schemes and game mechanics than the other sets.

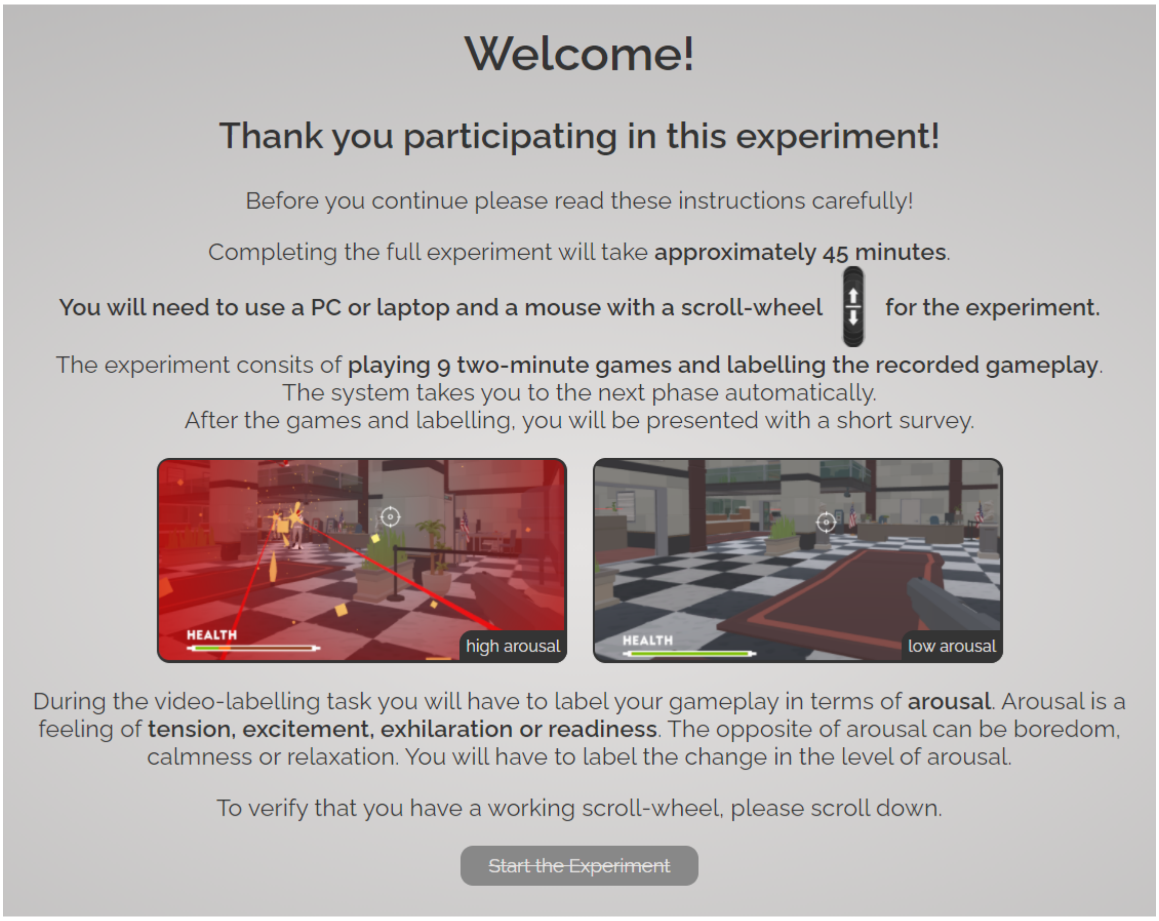
- **Endless** is a casual endless-runner game (see Fig. 4.1g). The player controls the character with just two keys and attacks with a third one. The goal of the game is to stay alive as long as possible on an endlessly looping level with random enemy and obstacle placement. The game moves forward (and accelerates) automatically; the player can only move between two predetermined tracks. The game also features a wide variety of pickups which can increase the player's score or increase and decrease the difficulty (making the game move faster or slower). The main challenge of the game is to avoid incoming enemies. On collision with an obstacle or an enemy, the player loses score, and the game's speed is reset to the initial value. The game was inspired by casual mobile games like *Temple Run* (Imangi Studios, 2013).

- **Pirates!** is a classical platformer (see Fig. 4.1h) which focuses on light traversal puzzles in a 2D environment. The player controls the character using three keys (two directions and one jumping). The game features one power-up, which gives the player a bonus life. Enemies can be eliminated by jumping on their heads; however, the player is respawned at the last checkpoint on a direct collision. The game was heavily inspired by *Super Mario Bros.* (Nintendo, 1985).

- **Run'N'Gun** is a shoot 'em up platformer (see Fig. 4.1i). The player controls the character with six keys (four-directional, one jump, and one attack). The game shares many resemblances with shooter games, as the player has a projectile weapon to fight enemies on the level, and the game features a health pack similarly to TopDown. Unlike shooter games, enemies come in a wider variety with both melee and ranged opponents and bosses with multiple health bars. Unlike other platformers in the set, players can earn a score in the game by defeating enemies. Similarly to Pirates!, if the player runs out of health, they are respawned at the last checkpoint. The game was inspired by *Metal Slug* (SNK, 1996).

4.3 Data Collection

The games were integrated into the PAGAN annotation platform (Melhart et al., 2019b), which allowed the large-scale crowd-sourcing of both the game playing and annotation tasks.



Welcome!

Thank you participating in this experiment!

Before you continue please read these instructions carefully!

Completing the full experiment will take **approximately 45 minutes**.

You will need to use a PC or laptop and a mouse with a scroll-wheel ↕ **for the experiment.**

The experiment consits of **playing 9 two-minute games and labelling the recorded gameplay**.
The system takes you to the next phase automatically.
After the games and labelling, you will be presented with a short survey.

During the video-labelling task you will have to label your gameplay in terms of **arousal**. Arousal is a feeling of **tension, excitement, exhilaration or readiness**. The opposite of arousal can be boredom, calmness or relaxation. You will have to label the change in the level of arousal.

To verify that you have a working scroll-wheel, please scroll down.

Figure 4.2: Introduction screen of the experiment.

4.3.1 Protocol

The game-playing and annotation task was crowd-sourced using Amazon's Mechanical Turk service [6]. To limit the noise in the dataset, only people with prior videogame purchase were invited for the experiment. Using the PAGAN platform, participants were able to play the games online from their browser. The games have been fully integrated into the platform transitioning seamlessly from the gameplay to the annotation task. Participants were greeted with short introduction and task description, which described *high arousal* as a feeling of *tension, excitement, exhilaration or readiness* and *low arousal* as a feeling of *boredom, calmness or relaxation* (see Figure 4.2). After the welcome message, participants were thrust into a randomly selected game. Each game took a maximum of 2 minutes to play. During their gameplay, telemetry logs and a video of their session were recorded. After the game-playing session, participants were automatically forwarded to the annotation task. After the annotation task had been completed, the participants were forwarded to the next randomly selected game. This cycle was repeated nine times until all games had been played. At the end of the experiment, a short exit survey recorded the biographical details of the participants. Data collection took between 45 to 44 minutes per participant, and each participant was compensated with 10 USD for their time and effort.

[6] `https://requester.mturk.com/`

4.3.2 Participants

The final dataset contains 124 participants, who played $1,116$ gameplay sessions and generated over 37 hours of video footage. The gender distribution of the dataset is somewhat skewed, with 1 participant identifying as non-binary, 43 as female, and 80 as male. The average age of the participants was 33 years (between 19 and 55). The majority of the respondents were from the USA (102 participants), while the rest were divided between Brazil (10 participants), Italy (3), Canada (2), India (2), Czech Republic (1), Germany (1), and Romania (1). Most of the participants (116) are self-described gamers and playing videogames daily or weekly. Regarding gaming habits, participants had either a PC, gaming console, or both and played a wide variety of games from casual and mobile games, through platformers, role-playing games, to sports simulators. The collected biographical data shows that the participants could have grasped the testbed games reasonably quickly.

4.3.3 Gameplay Footage

During the gameplay sessions, the participants' gameplay was recorded. These videos were used in the annotation task immediately after the gameplay but have also been published as part of the AGAIN dataset (Melhart et al., 2021) to aid future research into pixel-based affect modelling (Makantasis et al., 2019). The recorded gameplay footage amounts to over 37 hours of gameplay (more than 3×10^6 frames of video). The games were recorded at 24 FPS with a resolution of 960×600 pixels.

4.3.4 Game Telemetry

During the game-playing procedure, detailed telemetry has been recorded. Telemetry data could be viewed as domain-specific *privileged information* (Vapnik and Vashist, 2009), which encodes in-game events, player action and behaviour, and gameplay context. It generally consists of ad-hoc features based on how gameplay elements and mechanics are implemented. Fusing gameplay features with other user modalities has also been a dominant practice in game-based affective computing (Martínez and Yannakakis, 2014; Martinez et al., 2014); however, subsequent experiments in this study only rely on the game telemetry, discussed in this section.

All games implement the same data-logging strategy and use a similar method for recording telemetry. Games within the same genre share the same feature labels. These features are referred to as *genre-specific features* throughout the thesis. In addition to genre-specific features the dataset includes 13 *general gameplay features*. AGAIN provides 33, 35, and 42 genre-specific features for the racing, shooter, and platformer games, respectively, when redundancies between general and genre-specific features are eliminated. However, not all features have a qualitative meaning for all games within a genre—in Heist!, for instance, players move, but they are immobile in Shootout. To ease the data collection, aggregation, and later modelling process, when features are absent from a game, they are given values with zero-variance (zeroes or ones, depending on the feature). For example, a looping racetrack is only present in the Solid game (see Figure 4.1b); therefore, the Visible Loop Count feature is always zero in the other racing games.

The recorded genre-specific telemetry encodes *control events* initiated by the player (e.g. Player Steering), *player status* (e.g. Player Health), *gameplay events* outside of the player's control (e.g. Bot Aim at Player), *bot status* (e.g. Bot Off-road), and the *proximal* and *general game context* (e.g Bot Player Distance and Pickups Visible). The gameplay is

Table 4.2: The general gameplay features of AGAIN

feature	**description**
Time Passed	time counted from the start of the recording
Player Score	points indicating goal completion
Input Intensity	number of keypresses
Input Diversity	number of unique keypresses
Player Activity	ratio of time spent pressing keys
Player Movement	distance travelled + reticle moved (in shooters)
Bot Count	number of bots visible
Bot Movement	bot distance travelled
Bot Diversity	number of unique bots visible
Object Intensity	number of objects of interest
Object Diversity	number of unique objects
Event Intensity	number of events
Event Diversity	number of unique events

recorded at approximately 4Hz (every 250ms). Due to the limitations of the Unity engine and the WebGL format, the logging rate is not consistent. The logging script aggregates multiple ticks of the engine's update loop and provides an average value to mitigate this issue. Due to this processing technique, almost all events are represented by continuous values. For example, Pickups Visible can take float values under 1 when a pickup just became visible at the end of the given 250ms window. The only features represented by integer values are Player Death and Objects Destroyed because of their sparsity. Due to space constraints, genre-specific features are not listed here. Please read Appendix A.1 for a full list of genre-specific features.

General features are derived from the genre-specific telemetry. These general features are ad-hoc designed and are based on contemporary studies of general player modelling (Camilleri et al., 2017; Bonometti et al., 2020). Events that require expert evaluation of the game, such as the *goal-oriented* and *goal-opposed* events of Camillieri et al. (2017) are omitted from these general features of AGAIN but may be considered as additional features. Table 4.2 lists these features alongside their explanation.

4.3.5 Annotation

The annotation task was administered through the PAGAN platform (Melhart et al., 2019b), using the *RankTrace* annotation method (Lopes et al., 2017b) (see Chapter 3). *RankTrace* allowed for the collection of data in an unbounded fashion (see Fig. 4.3). This type of data is best interpreted as subjective, ordinal labels as it preserves the relative relationships between data points (Yannakakis et al., 2018). The unbounded trace means that users can always adjust their annotations higher or lower than previous values, which alleviates much of the guesswork compared to when users annotate on an absolute and objective scale (Martinez et al., 2014). The ordinal nature of the annotation follows the cognitive process of human evaluation, as it provides a trace of which factors in habituation (Solomon and Corbit, 1974), anchoring bias (Damasio, 1994; Seymour and McClure, 2008) and recency-effects (Erk et al., 2003).

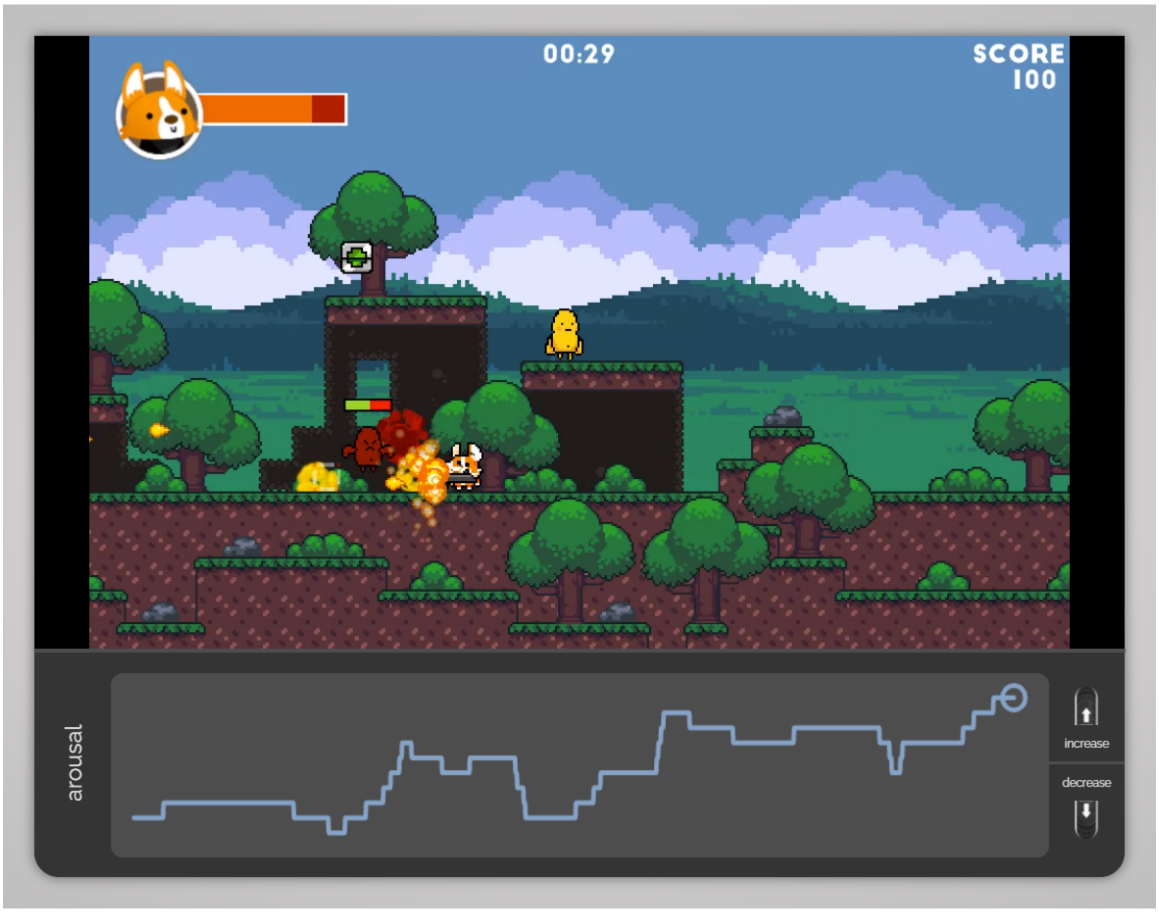


Figure 4.3: The PAGAN *RankTrace* annotation interface. The gameplay video is played in the window above and the participant controls the annotation cursor (blue circle) below, drawing a visible annotation trace.

4.3.6 Data Cleaning

To ease subsequent analysis and studies, the dataset is cleaned of unresponsive participants and clear outliers. Through a multi-step process, 10.8% of the data points are removed. The published AGAIN dataset (Melhart et al., 2021) contains both the raw and the cleaned data that result from the process outlined here.

Since PAGAN only records annotations when there is a change in the signal and the Unity engine loop is affected by hardware performance, as a first step, the dataset is resampled at 4Hz to get a consistent signal. Duplicate values are removed from the dataset and sessions that are either too short (less than 1 minute) or too long (more than 3 minutes) due to software or technical errors during crowd-sourcing. Sessions that have less than 10 annotation points are pruned, assuming that the participant was unresponsive. This initial cleanup phase removes 24 sessions (2.1% of the data).

To clean the dataset further, Dynamic Time Warping (DTW) is applied to get an approximate similarity measure between traces. DTW is used in time-series analysis to measure the similarity between temporal sequences that might be out of sync or vary in speed (Berndt and Clifford, 1994). DTW works by calculating a warping path between two signals using a similarity matrix. It provides a valuable metric that qualifies time-series

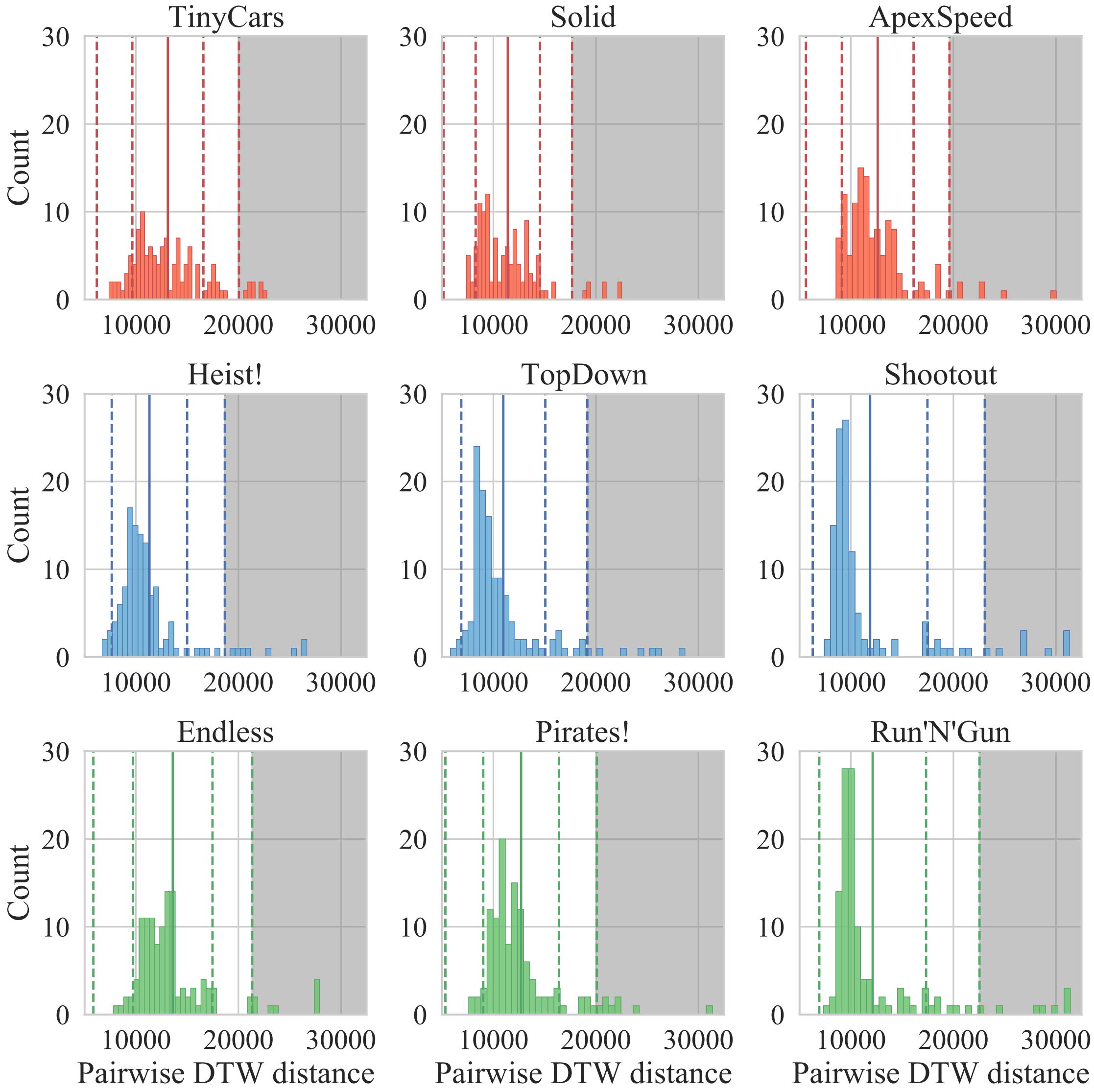


Figure 4.4: Distribution of summed cumulative DTW distance values of each session compared to every other session. The solid line shows the average score, while the dotted lines show the first and second standard deviation. Values in the grey field (right tail) are removed during data cleaning.

in the form of a cumulative DTW distance describing the similarity to a baseline trace or other signals (Berndt and Clifford, 1994). Both of these strategies are applied when cleaning the dataset. As a first step, the cumulative DTW distance to an artificial flat baseline is calculated (arousal annotations at 0 in all time-windows). The resulting score provides a similarity measure to an artificial session where the participant performed no annotation; this allows the removal of unresponsive outliers. Sessions that fall more than two standard deviations closer to zero from the average cumulative distance (the left tail of the distribution) are removed. This step removes 28 additional sessions from the dataset (2.5%). Finally, the cumulative DTW distance metric is applied between each data point,

Table 4.3: Preliminary analysis of the clean AGAIN dataset. The table lists the number of game sessions and their corresponding data points on a frame-by-frame basis (250 ms). The table also lists the number of 3s time-windows within which the arousal value increases (↑), decreases (↓) or stays stable within a 10% threshold bound (—).

			Arousal (3 s interval)		
Game	**Sessions**	**Data** ($\cdot 10^3$)	↑	↓	—
TinyCars	109	52.75	543	461	3386
Solid	109	53.42	613	492	3346
ApexSpeed	114	56.10	607	462	3581
Racing	332	162.27	1763	1415	10313
Heist!	110	53.91	580	424	3479
TopDown	115	56.90	650	463	3614
Shootout	106	51.77	471	341	3496
Shooter	331	162.57	1701	1228	10589
Endless	112	55.11	559	438	3595
Pirates!	110	52.26	625	534	3186
Run'N'Gun	110	54.97	618	431	3521
Platformer	332	162.34	1802	1403	10302
Total	995	487.18	5266	4046	31204

and the resulting distances are summed up. This metric shows the relative similarity of a session to every other session. All sessions which fall more than two standard deviations away from the average summed cumulative distance (see Fig. 4.4) are removed. This step removes an additional 69 sessions (6.2%), which are too dissimilar from the general trends of the participants' annotations; it is presumed that either the annotation was improper or that this session's elicitor was somehow not in line with how other players played the same game. At the end of the cleaning process, 121 sessions—including all data from 2 participants—are removed (10.8%). The cleaned dataset consists of 122 participants, 995 sessions, and 490,494 data points.

4.4 Preliminary Analysis

This section presents a preliminary analysis of the collected data through a statistical lens. The purpose of this investigation is to contextualise and help to explain the results of the subsequent modelling efforts. The chapter highlights similarities and differences between games and genres and examines input and output features, their relationship, and their implications to the modelling tasks ahead. The first half of this section focuses on general trends in the data, while the latter half focuses on correlation analysis between the features and the annotated ground truth. To better connect the analysis to the subsequent modelling tasks, the data is pre-processed for machine learning before the correlation analysis.

4.4.1 General Trends

Following the cleanup process presented in Section 4.3.6, this section performs a preliminary analysis of the clean version of the AGAIN dataset focusing on patterns in the arousal

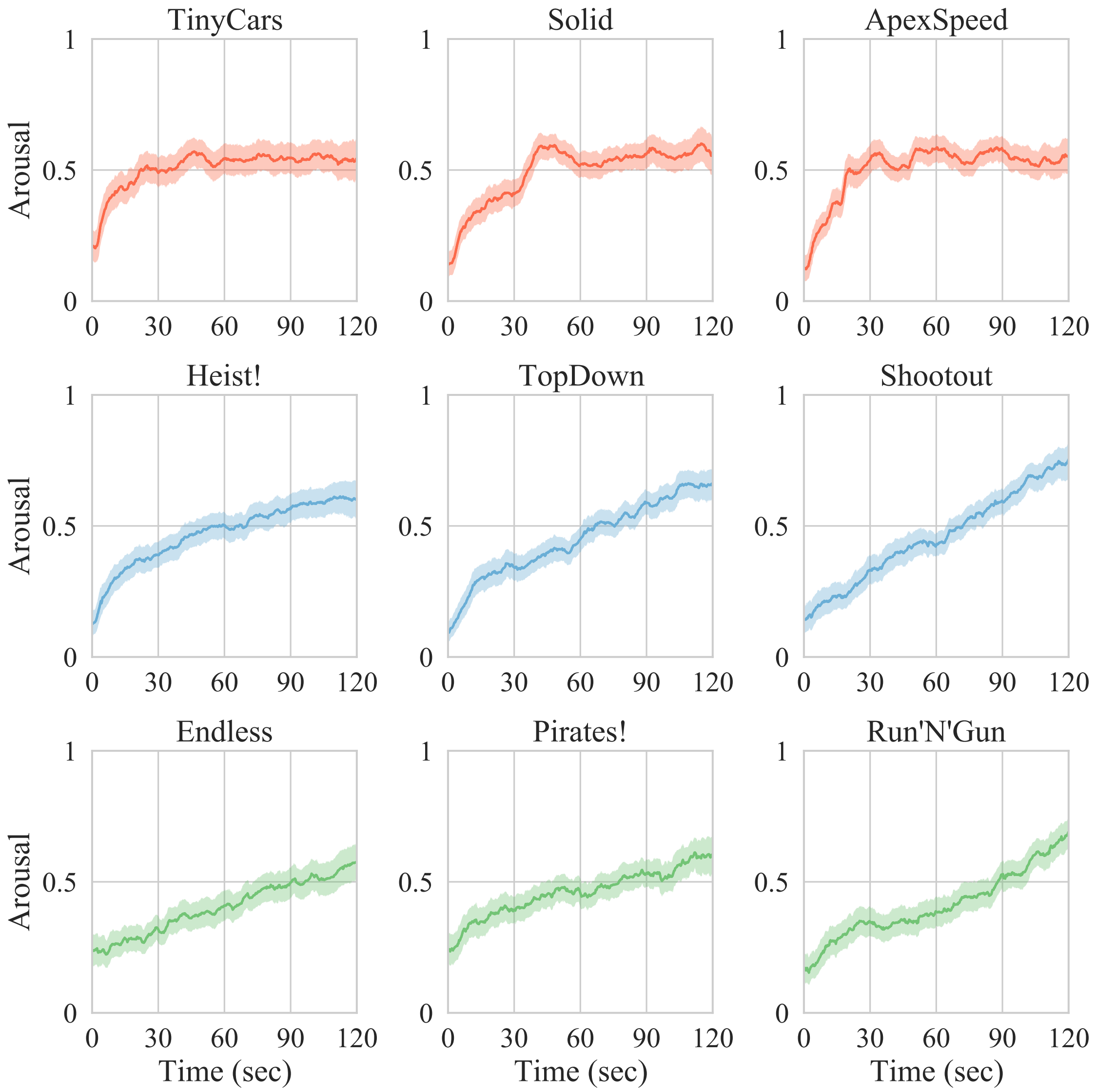


Figure 4.5: Average annotation traces (normalised per session) showing an increasing tendency. The coloured area around the mean depicts the 95% confidence interval of the mean.

annotations and the AGAIN game context features. While some games receive more aggressive data cleaning than others (*TinyCars, Solid,* and Shootout), overall, there is an even distribution of data and sessions across genres as shown in Table 4.3.

Trends in the Annotations

Figure 4.5 shows the average annotation trace as calculated by averaging values in time-windows of 250 ms of all sessions' traces. It is evident that arousal annotation tends to have an upward tendency. This is not surprising, as most games considered are action-oriented with an ever-increasing challenge; for instance, Endless keeps increasing the game's speed, which makes it both more challenging and more arousing as time passes. Racing games (top row of Figure 4.5), on the other hand, tend to have arousal converging to a maximum mean value after the first 30 seconds. This is likely because the player is initially rushing to overtake the opponents' cars (players always start last); after this initial excitement, the race becomes repetitive, with players trying to either maintain the lead or slowly catch up to the leader.

Trends in the Game Context Features

Observing the twelve general gameplay features shared across all nine games, one can detect notable differences between games. In terms of the player's input (control), games with more complex interaction schemes appear to have higher input diversity and input intensity (see Table 4.2 for details on these features). Even accounting for the games' different control schemes (i.e. the number of controls the player has available), ApexSpeed, Shootout, and Endless have the lowest intensity (number of keypresses) and diversity (number of unique keypresses) while Pirates! and TinyCars have the highest diversity. This discrepancy could point to a more manageable control scheme for the former games, but it could also point to a more frantic and engaging interaction in the latter games. The idle time and activity features corroborate this observation, as racing games have less idle time without keypresses (since in two of the games the player needs to press a button constantly to move forward). In contrast, games where participants mainly reacted to stimuli (e.g. in Shootout, players react to opponents popping up and in Endless players jump only when a gap or obstacle is near) featured much higher idle times. In terms of other features, the number of bots (opponents) visible on the screen varied wildly between games, with Tiny Cars and Shootout having the highest number of visible enemies on average. Perhaps due to the many enemies present, Shootout had the highest number of events (Event Intensity on Table 4.2), while Solid had the fewest events per time-window.

In terms of comparing the general gameplay features across games, this requires some normalisation to account for both discrepancies in value ranges between games (e.g. in terms of score) and between players in the same game. Following the paradigm of treating both input and output as relative (Camilleri et al., 2017), the gameplay features of each time-window are normalised to the $[0, 1]$ value range within each session. As a result, such normalised features consider the dynamics of a single player in a given session (e.g. in which time-window the player achieved the top score of their session), disregarding, for example, whether other players reached higher scores in the same game. Since arousal is similarly a deeply subjective notion, the player is expected to annotate arousal in the context of their current session (e.g. whether their arousal might increase if they start performing better than they were performing previously in the same session). After all $4.9 \cdot 10^5$ game context data was normalised in this fashion, we applied t-distributed Stochastic Neighbor Embedding (t-SNE) (van der Maaten and Hinton, 2008) to map this data on a two-dimensional space. Figure 4.6 shows the resulting data distributions. The visualised distributions offer some important insights into the differences between games. In particular, every game's general features tend to exhibit different patterns than the other eight games. Moreover, the compressed (game context) feature distributions across the three shooter games (see the middle row of the figure) appear quite distinct from one another. In some cases, however, there appears to be an overlap between games of the same genre (e.g. all racing games) or games of different genres (e.g. see Pirates! and Solid). Even though this type of data visualisation cannot shed light on all possible differences between games, it indicates that the games impact the patterns of the data solicited from players (i.e. the context influences the user behaviour). The t-SNE analysis also indicates that mapping

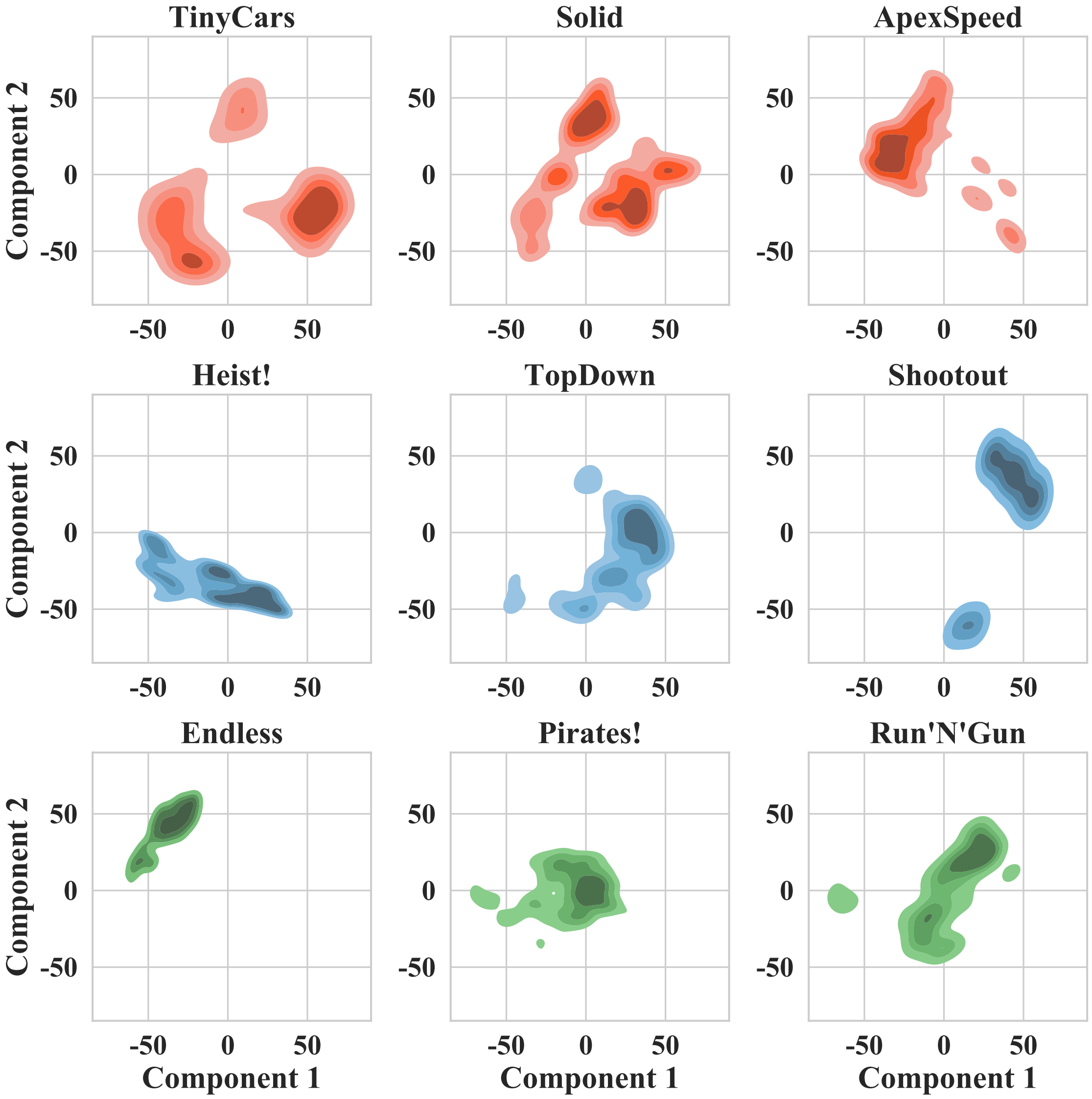


Figure 4.6: Projections of general game features in a 2D space with t-distributed Stochastic Neighbour Embedding.

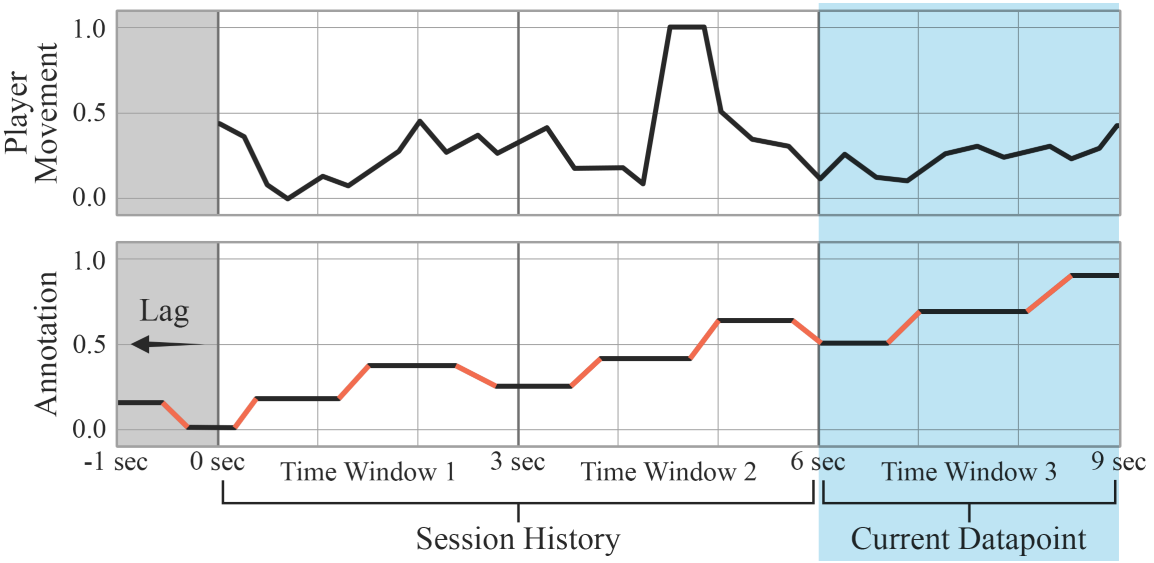


Figure 4.7: The aggregation of 3-second time-windows. The figure shows the application of the 1-second lag on the annotation and an example of comparing the highlighted Time Window 3 to the average of the Session History (Time Window 1 and Time Window 2). The red highlights on the annotation signal show the segments which affect the calculation of the average gradient. Note that although the mean values of Time Window 1 and Time Window 2 are very different, their average gradient is very similar.

between game content and arousal seems to be easier for some games (and game genres) than others.

4.4.2 Pre-Processing

Although the dataset was recorded at 4Hz, windows of 250ms are not meaningful intervals in terms of human attention due to reaction time. Subsequently, the data is processed into 3-second time-windows using two distinct metrics. Inspired by the studies of Lopes et al. (2017b) and Camilleri et al. (2017) arousal is processed firstly as the *mean* (μA) trace of a selected time-window. A preference learning model trained on this ground truth predicts the ordinal change in the level of player arousal. Secondly, arousal is processed as the *average gradient* (∇A) of the trace within a selected window (see Figure 4.7). Preference learning models trained on this ground truth predict the change in the acceleration of player arousal. While this metric is less intuitive compared to μA, it gives an estimation of the temporal dynamics of the player experience as it registers whenever a linear trend in arousal changes. Because of this focus on the change in dynamics instead of the change in absolute values, ∇A should also provide a trace that is more robust against recency effects such as *habituation* (Yannakakis et al., 2018). Following standard practice in affective computing (Metallinou and Narayanan, 2013; Lopes et al., 2017b; Camilleri et al., 2017), a time offset of 1-second is also introduced to the annotation traces. Mariooryad and Busso suggest that although an optimal annotation lag can be found algorithmically, an ad-hoc value between 1 and 3 seconds is practically a good compromise when it comes to similar annotation tasks (Mariooryad and Busso, 2013). Through a preliminary experiment (set up similarly to experiments described in Chapter 5), it has been determined that an

Table 4.4: Top five Kendall's τ correlations between input features and the output processed as the *mean* (μA) and *average gradient* (∇A) of time-windows. Features marked with G are general, while features marked with S are genre-specific features.

	μA			∇A		
Genre	**Feature**		τ	**Feature**		τ
Racing	G	Player Score	0.166	S	Player Lap	-0.069
	G	Time Passed	0.163	G	Time Passed	-0.068
	S	Player Lap	0.125	G	Bot Count	0.066
	G	Bot Diversity	-0.097	G	Bot Diversity	0.065
	S	Player Speed	0.093	S	Bot Grounded	0.065
Shooter	G	Time Passed	0.363	S	Bot Proj. Count	0.129
	G	Player Score	0.338	S	Bot Proj. Player Dist.	0.119
	S	Bot Aim at Player	0.158	S	Bot Shooting	0.113
	G	Event Diversity	0.151	S	Player Damaged	0.089
	S	Bot Shooting	0.148	S	Bot Rotation	0.085
Platformer	G	Time Passed	0.271	S	Player Health	0.120
	G	Player Score	0.269	G	Bot Movement	0.099
	S	Bot Proj. Count	0.114	S	Player Pickup (Boost)	0.097
	S	Bot Proj. Player Dis.	0.113	G	Bot Diversity	0.079
	G	Bot Diversity	0.106	G	Bot Count	0.074

annotation lag of 1 second is sufficient to correct for the participants' reaction. As mentioned above, to aid the subsequent modelling tasks, the dataset is normalised on a *per session* basis. Figure 4.7 illustrates the relationship between the input features and the measured ground truth in the dataset. After this 1-second lag correction, the dataset consists of 40836 datapoints. As discussed in Section 2.3.3, during the subsequent modelling tasks, the number of observations can vary depending on how many valid comparisons can be found during the pairwise transformation. Figure 4.7 also illustrates how the dynamic memory windowing is applied when the full session history is considered (for more details on the method see Section 2.3.3).

4.4.3 Correlation Analysis

Before concluding this chapter and moving on to preference learning models, this section presents the results of a correlation analysis between individual input features and the two different outputs discussed in Section 4.4.2. This type of analysis highlights the linear feature importance, which can help explain some of the non-linear modelling results in subsequent chapters. Correlations are measured using Kendall's τ, which is a monotonic measure of rank-correlation that has a great degree of robustness against outliers (Nelsen, 2001). Because of this robustness to noise and its applicability to non-parametric problems and ordinal data, Kendall's τ handles human-generated data (such as annotations) very well. Significance is reported at a 95% confidence interval and adjusted using the Bonferroni correction where applicable. Due to space constraints, the complete list of correlations is listed in Appendix A.2.

Table 4.4 shows the top five correlations between and input features and both μA and ∇A for each genre discussed in this thesis. It is evident that the output processed as

μA have stronger correlations with the input features. Looking at Figure 4.5, it is not surprising that the top two features in each genre are Time Passed and Player Score. Player Score also correlates with time in all games and hence can be considered a time-related feature. Evidently, the μA processing of arousal can capture the temporal dynamics of the presented games in a linear fashion to a good degree. While this dynamic might not be true to all modern games, as the increase in arousal would hit a diminishing return (also see the racing games on Figure 4.5 for an example), games in the dataset built similarly to casual and mobile games, replicating fast-paced but short gameplay sessions more typical to these domains. However, since time—which is a game-agnostic feature—is already a good linear predictor of arousal across all games, it is expected that μA models will perform and generalise well. Table 4.4 shows that this connection is the strongest in shooter games, followed by platformers, and finally by racing games. This finding already foreshadows the difficulty of the modelling task for each of these genres. General features have strong correlations within each genre, however. When looking at the genre-specific features, an interesting finding is the high correlation of Bot Projectile Count and Bot Projectile Player Distance in platformer games which are features used only in Run'N'Gun. In contrast to these results, ∇A show weaker correlations with the input features. While Time Passed is still a prominent feature in racing games, it is missing from other games. Player Score, another feature highly correlated with μA is also missing from the top five correlations of ∇A. While there is some overlap between the two outputs, genre-specific features are much more prominent. This is true, especially in shooter games, where all the top features are genre-specific. While with μA the most robust features were time-related, with ∇A most of these features are related to the bots and in-game events except for racing games. The diversity of the top features and their relatively weak correlation with arousal foreshadows a more challenging task that is less linearly separable compared to μA.

4.5 Summary

The first half of this chapter detailed the AGAIN dataset, which was collected for the subsequent studies in general affect modelling. In particular, the motivation for the dataset was described, identifying a need for comprehensive and diverse game datasets for general affective player modelling research. The bulk of the chapter described the produced games, the collection process, and the different attributes of the dataset. Particular focus was given to easily identifiable general features, which will be used in subsequent studies. The later half of this chapter presented a preliminary analysis of the AGAIN dataset. This overview aimed to uncover some underlying patterns and connections that can help contextualise and interpret subsequent machine learning results. Section 4.4.1 a qualitative analysis of emerging trends in the dataset regarding the annotation values and the distribution of general features between games. Section 4.4.2 explained the pre-processing steps, which are necessary for future machine learning tasks, and finally, Section 4.4.3 presented an analysis of the linear connections between input and output features. This investigation uncovered that when annotation traces are processed based on the mean value of time-windows, the resulting output feature significantly correlates with time-related features.

The next chapter presents the preliminary results of game-based affect modelling with game-agnostic features. The first sections focus on model validation and modelling results, while the latter sections reevaluate the models in light of feature importance and time-windowing technique.

Chapter 5

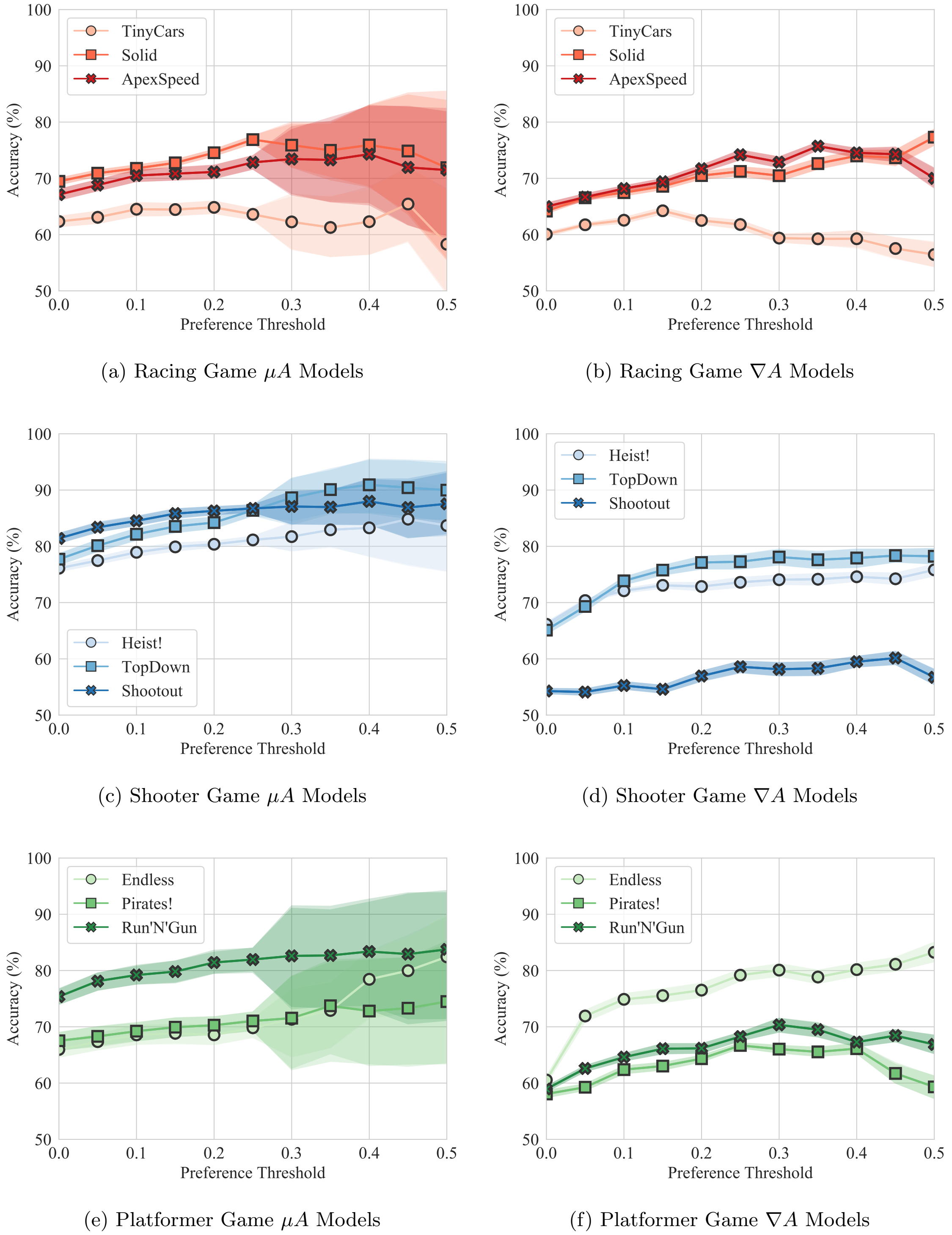
Game-Based Arousal Modelling

This chapter presents the preliminary modelling results focusing on models trained and tested on single games. The primary purpose of these models is two-fold. First, to show the effectiveness of a general feature set compared to a genre-specific one. Second, to provide a baseline for the subsequent experiments. Studies presented throughout this paper rely on ordinal data processing and modelling, in accordance with contemporary research that highlights the ordinal nature of human emotions and cognitive processes (Yannakakis et al., 2018). A growing body of research is dedicated to the ordinal processing and modelling of emotions—not merely in game research but in affective computing as a whole (Yannakakis and Martínez, 2015; Yannakakis and Martinez, 2015; Lopes et al., 2017a; Camilleri et al., 2017; Melhart et al., 2020b; Yannakakis et al., 2018). These studies show that beyond a *first-order* ordinal representation (where datapoints are already captured in a relative fashion), a *second-order* processing (where datapoints captured as absolute values are translated into an ordinal representation) improves the reliability and validity of the derived models (Yannakakis et al., 2018). Studies presented in this thesis continue this line of inquiry by applying preference learning methods to create ordinal affect models. Namely, a *pairwise transformation* (introduced in Section 2.3.3) is applied to the dataset prior to modelling. The goal of this transformation to convert the data into an ordinal representation. Throughout this chapter results with annotations based on the *mean* (μA) and *average gradient* (∇A) trace of a selected time-window are discussed separately (see Section 4.4.2 for more information on these metrics). As predicted during the correlation analysis of the features in Section 4.4.3, the results obtained show that the ∇A task is considerably harder than μA.

This chapter is structured as follows: Section 5.1 discusses the considered hyperparameters and the tuning process. Section 5.2 shows the results of models trained on genre-specific and general features within individual games. Section 5.3 investigates the non-linear feature importance of the input of the models. Before concluding the chapter, Section 5.4 reflects on the effect of the dynamic memory window, applied during the pairwise transformation, on model performance. Finally, Section 5.5 gives a summary of the chapter.

5.1 Model Validation and Parameter Tuning

As described in Section 2.3.3 studies in this thesis use preference learning via Random Forests (RF) for predictive modelling. Because RFs are stochastic methods, all experiments throughout the thesis are repeated 20 times, and the average of all runs is reported. To test the results of each model, 10-fold cross-validation is used. The cross-validation folds



(a) Racing Game μA Models

(b) Racing Game ∇A Models

(c) Shooter Game μA Models

(d) Shooter Game ∇A Models

(e) Platformer Game μA Models

(f) Platformer Game ∇A Models

Figure 5.1: Performance of μA and ∇A models trained with different Preference Thresholds (P_t). Figures on the left show μA models, while figures on the right show corresponding ∇A models. The coloured bands indicate a 95% confidence interval.

Table 5.1: Information loss during the pairwise comparison due to an increased P_t compared to $P_t = 0$. The bold values show the loss corresponding to the selected P_t values for subsequent models.

P_t	**Data Lost (%)**	
	μA	∇A
0.05	14±2	**35±3**
0.1	28±2	53±2
0.15	**40±2**	66±2
0.2	51±2	74±2
0.25	61±2	81±1
0.3	69±1	86±1
0.35	76±1	89±1
0.4	82±1	92±0
0.45	88±1	94±0
0.5	91±1	96±0

are defined between subjects. Because 122 subjects cannot be divided evenly, each fold encompasses either 12 or 13 players. The same cross-validation strategy is maintained with both game-based models in this chapter and general models in subsequent chapters to make the models comparable. This means that models are always predicting unseen players. The reported statistical significance throughout the thesis is measured with two-tailed Student's t-tests with 95% confidence interval, adjusted with the Bonferroni correction where applicable.

Parameter tuning focuses primarily on the *Preference Threshold* (P_t) of the pairwise transformation (see Section 2.3.3). In particular, the best P_t value parameter is sought between 0 and 0.5 with steps of 0.05. Figure 5.1 shows the results of the parameter tuning. As is evidenced by these results, increasing P_t leads to an increase in performance in most cases. Similarly to an uncertainty bound in classification, increasing P_t reduces the noise in the dataset as higher P_t values filter out points that fall closer together. As the dataset is normalised on a session level, a $P_t = 0.05$ means that pairs of points are only considered if they have at least 5% difference during the pairwise transformation. While the P_t is useful in reducing the surface level noise of the data, there are some caveats to the method. As it is shown on Figure 5.1, μA models tend to become unstable with P_t values above 0.3. This phenomenon could be caused by high P_t values getting rid of useful datapoints as well as noisy ones, decreasing the robustness of the models. This trade-off between noise reduction and information loss could also decrease the reliability of models. As it is shown on Table 5.1, compared to $P_t = 0$—that is only excluding datapoints with the same outputs from the pairwise transformation—increasing P_t leads to a rapid information loss. To contain the bias of the models, the best P_t is picked, given that at least 50% of the available comparisons is maintained. Given this constraint, the extensive empirical experiments presented here show that $P_t = 0.15$ and $P_t = 0.05$ yield the most accurate models—for μA and ∇A, respectively.

As discussed in Section 2.3.3, a dynamic windowing method is introduced as an additional method to control for temporal effects. This method aims to naively approximate the *memory* of a player as it pertains to previous time-windows. During the *pairwise transformation* each consecutive datapoint is compared to the average of all previous datapoints

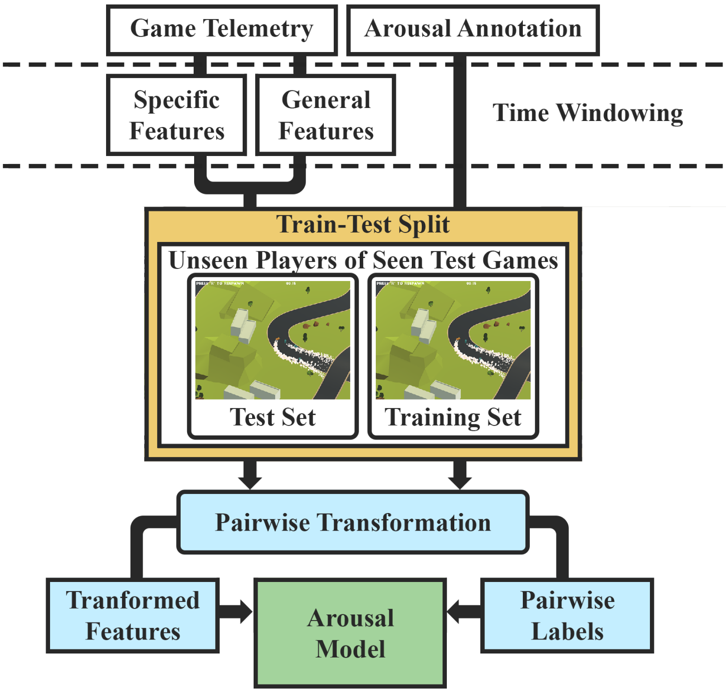


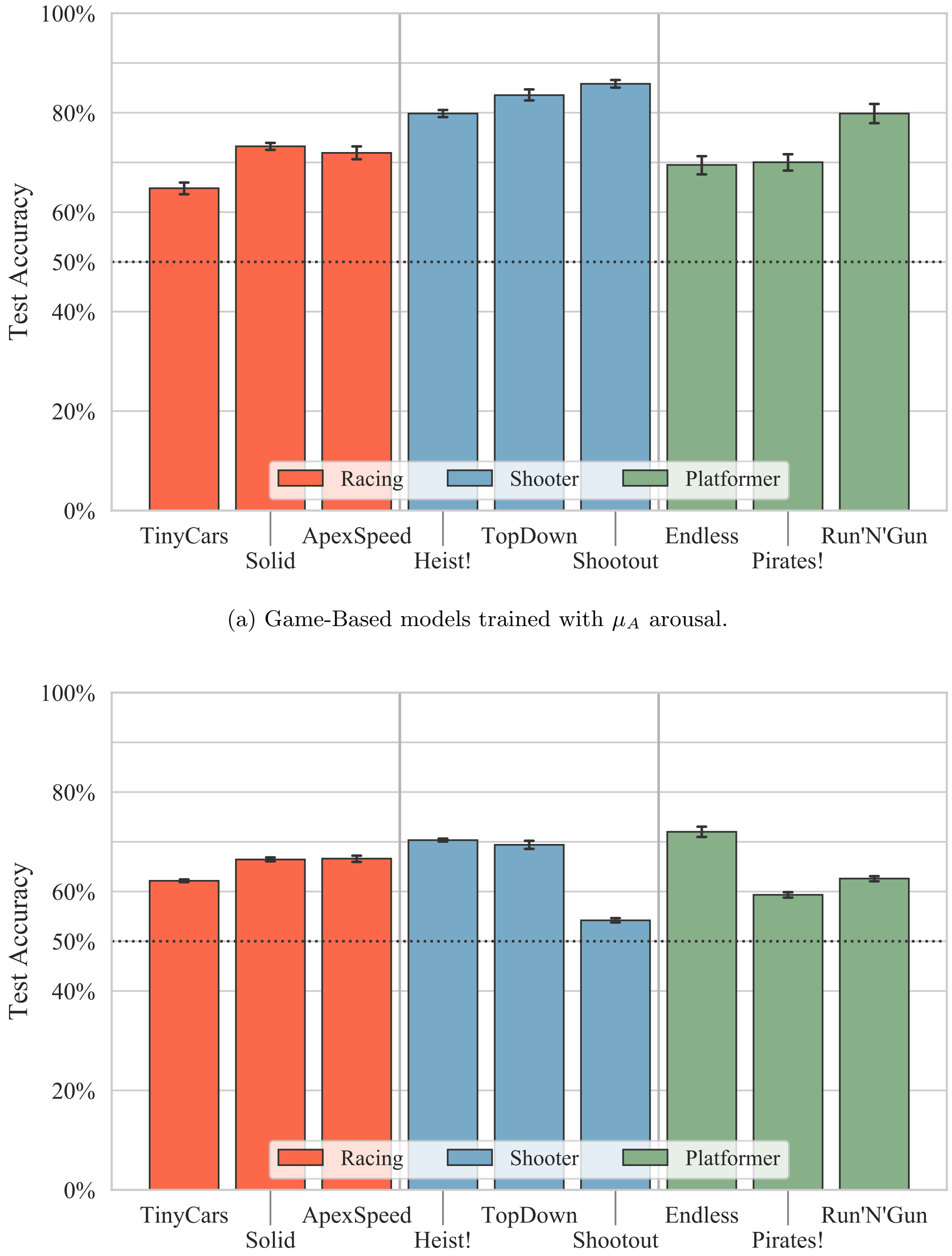
Figure 5.2: Game-based modelling pipeline for modelling arousal. Both genre-specific and general features are extracted from the raw telemetry. Preference learning is applied by using a Pairwise Transformation, in which the ranking problem is reformulated as binary classification of pairwise labels (see Section 2.3 for more details).

in the session. Models using this windowing method potentially encode more information about past gameplay events. If not indicated otherwise, all models use this type of dynamic windowing technique. Section 5.4 reexamines this parameter and its impact on the performance of the models.

5.2 Game-Based Model Performance

Figure 5.3 gives an overview of the best results reported in this section, while Table 5.2 shows a more detailed account of the test accuracies of models trained and tested on the same games. To measure the robustness of general features, genre-specific and general feature sets are compared to each other and models using all available features. Game-based μA models have the highest accuracies when the models are trained on a combined feature set of genre-specific and general features in 5 out of 9 cases. Game-based ∇A models have the highest accuracies when the models are trained on a combined feature set of genre-specific and general features in 7 out of 9 cases. Notably, with μA models, shooter and platformer games benefit from these combined feature sets, while in ∇A this trend is true to all games except for TinyCars and Shootout. Interestingly, with μA models, most racing games models based on general features outperform models based on both the combined and genre-specific feature set. While the overall general features lead to better predictions than specific features, in the case of μA models, there are some exceptions (see TinyCars, Solid, and Endless).

With game-based μA models, differences between models with different inputs are not



(a) Game-Based models trained with μ_A arousal.

(b) Game-Based models trained with ∇A arousal.

Figure 5.3: Overview of the performance of game-based models trained on the best feature set for each game in the dataset. Models are trained and tested on the same games (indicated by the *x-axis*). The dotted line shows the natural baseline and the error bars indicate a 95% confidence interval.

significant—except for Solid, where models trained on general features significantly outperform ones that were trained on genre-specific features. The best performing μA models are

Table 5.2: Testing accuracies (%) of game-based models trained and tested on the same game. Bold values show the best models for the given game and target output.

μA			
Features	**Specific**	**General**	**All**
TinyCars	**64.8±1.2**	64.3±0.9	64.4±1.1
Solid	71.8±0.4	**73.2±0.7**	72.7±0.6
ApexSpeed	70.5±1.1	**71.9±1.3**	70.8±1.1
Heist!	79.4±0.6	79.4±0.7	**79.8±0.7**
TopDown	82.8±1.1	83.3±1.1	**83.5±1.1**
Shootout	85.8±0.8	85.8±0.8	**85.8±0.8**
Endless	**69.5±1.8**	69.1±1.8	68.9±1.7
Pirates!	69.5±1.6	68.9±1.7	**70.0±1.7**
Run'N'Gun	79.5±1.8	79.8±1.9	**79.8±1.9**

∇A			
Features	**Specific**	**General**	**All**
TinyCars	**62.2±0.3**	60.2±0.3	61.8±0.3
Solid	65.9±0.4	63.6±0.4	**66.4±0.4**
ApexSpeed	66.5±0.6	65.7±0.6	**66.6±0.6**
Heist!	69.5±0.3	69.5±0.3	**70.3±0.3**
TopDown	69.0±0.8	67.3±0.7	**69.4±0.8**
Shootout	**54.2±0.4**	52.5±0.3	54.0±0.5
Endless	71.9±1.1	71.1±0.9	**72.0±1.0**
Pirates!	58.7±0.5	59.1±0.5	**59.3±0.5**
Run'N'Gun	59.5±0.5	61.8±0.5	**62.6±0.5**

trained and tested on shooter games (with an average accuracy of 83%) followed by the platformers then racing games (with an average accuracy of 73% and 70% respectively). With game-based ∇A models, there are more significant differences between different feature sets. Except for ApexSpeed, Endless, and Pirates!, all games have a significant performance increase between their worst and best feature sets. However, ∇A models in all genres are performing at 65% average accuracy. It is worth noting that the performance of models trained and tested on Shootout is significantly worse than other shooter games, dragging down the genre average.

The lack of significant difference between feature sets among μA models shows the robustness of general features in capturing the complexity of gameplay within each genre. On the other hand, the lack of significant performance increase when combining the feature sets is possibly due to redundancies between genre-specific event telemetry and features such as Event Intensity and Event Diversity (see Table 4.2), which accumulate gameplay events. Conversely, the significant performance difference between features sets among ∇A models show that genre-specific features are not redundant when it comes to capturing a more complex temporal dynamics of the player experience beyond the absolute change in arousal. This is also supported by the fact that in most cases, the best feature set for ∇A models were the combined set of both genre-specific and general features.

Comparing the performance on the two tasks, it is evident that ∇A is a more complex task than μA. As discussed in Chapter 4, ∇A correlates more with genre-specific features

Table 5.3: Feature importance as derived from the Random Forests, averaged across games of the same genre. Features are labelled as general (G) or specific (S). Features present in top five features of all models with the same output are shown in bold.

Genre	μA			∇A		
		Feature	**Score**		**Feature**	**Score**
Racing	G	**Time Passed**	0.089	S	Player Collision	0.036
	G	**Player Score**	0.085	G	Time Passed	0.036
	S	Player Gas Pedal	0.062	G	Player Score	0.035
	G	Player Activity	0.045	S	Player Speed	0.035
	S	Bot Score	0.033	S	Player-Way Point Dist.	0.033
Shooter	G	**Time Passed**	0.167	S	Bot Shooting	0.056
	G	**Player Score**	0.126	S	Bot Health	0.046
	S	Bot Health	0.054	G	Bot Diversity	0.046
	G	Bot Count	0.051	G	Bot Count	0.043
	G	Bot Diversity	0.050	S	Player Damaged	0.042
Platformer	G	**Time Passed**	0.106	G	Bot Movement	0.043
	G	**Player Score**	0.104	S	Bot Speed X	0.038
	S	Player Damaged	0.039	G	Bot Count	0.037
	G	Bot Movement	0.037	S	Player Health	0.036
	G	Player Movement	0.035	S	Player Speed X	0.031

that are largely time-agnostic. It was hypothesised during this correlation analysis that ∇A would pose a harder problem as it has a weaker connection to general gameplay features. This observation is reaffirmed by the results presented in this section as well. While μA models trained on different feature sets generally have no significant differences, making models trained on general features a viable option with little to no trade-off, ∇A models often do significantly worse on general features alone.

The presented results also highlight some challenges for future research in general. The differences in performances between games show that the complexity of the affect modelling task is dependent on the characteristics of the elicitor and the game context. It is unlikely that general models will work equally well on all games. Finding new processing methods, data treatment, algorithms, and model architectures that perform equally well across different games is an open problem. However, the robustness demonstrated by the μA models trained on general features in section 4.3.4 point towards the possibility of general affect modelling across games. While research has already begun investigating general affect modelling in videogames, early results showed only moderate success. Subsequent chapters will focus on investigating the robustness of these general features in genre-specific and truly general setups.

5.3 Impact of Features on Model Performance

To better understand our results and the reason behind the unexpected robustness of general features, the top five most important features are observed in every genre. Here, the *Mean Decrease Impurity* (MDI) values of features are averaged from different training folds and within a genre to get a bigger picture (read more on MDI in Section 2.3). For a detailed breakdown of all features across game-based models, see the tables in Appendix A.3.1.

Because there was no significant difference between the models trained on different feature sets in the case of μA models, and in the case of ∇A, the best models were generally based on a combined feature set of all features, this section uses models that are trained on all (specific and general) features. This also has the additional benefit to maximise the number of features observed.

Table 5.3 shows the top five features in each genre ranked by their MDI values. Across all μA models, Time Passed and Player Score are the most important features. Because Player Score is generally increasing as the games progress, just like Time Passed, it is also a time-related feature. The prominence of these features across the board explains the robustness of general features presented in Section 5.2. The importance of time makes sense in the context of the games included in AGAIN as they are all intended to be casual and arcade-like. Games like these are generally designed with an ever-increasing intensity. When it comes to ∇A models, genre-specific features are much more prominent. Unlike μA models, where the top two features are general features and given considerably more weight, ∇A models rely on multiple features more evenly, and the most prominent features are often genre-specific. In these models, time-related features are given less weight, to the point where Time Passed and Player Score are only prominent in racing game models.

Analysing Table 5.3 by genre, it can be seen that features relating to player action are more important in racing games. This makes sense as the competition in these games is based more on the individual's skill than adversary play. In many cases, the player swiftly overtakes the bots (or is left behind), limiting their interaction. In shooter games, both μA and ∇A models focus more on the bots and their avatars' health. In platformer games, the player's status and the presence and movement of bots is more prominent. Unsurprisingly, the health of the bots (important for shooting games) is replaced with the movement of the bots as anticipating the bots' position is important for winning in these games. In the case of ∇A models, even more weight is put on horizontal movement (both for the player and the bots), which is typical of 2D platformer games. It is worth noting that the best models presented in Section 5.2 (μA shooter game models) all rely on time-related features to an exceptional degree.

Compared to the linear correlation between the input and output features presented in Table 4.4 in Chapter 4, while some features gain more prominence through non-linear modelling, in μA models, the top features remain the same (Time Passed and Player Score). In contrast to the correlation analysis, platformer models rely on some additional features present in all platform games. Racing game ∇A models rely more on player-specific feautres than anticipated, and shooter game models focus more on the bots' status rather than on projectiles. This is surprising as these features were among the top raking correlations in Chapter 4. Finally, ∇A platformer models rely on horizontal speed rather than Bot Diversity and Player Pickups, which were predicted by the correlation analysis.

5.4 Sensitivity to Memory

This section revisits the results of the previously constructed models to observe the effects of dynamic memory windowing on model performance. To provide a complete picture of how changing the memory from consecutive time-windows to the available session history (see Figure 2.3 in Chapter 2) impacts performance in terms of accuracy, this section compares the best models presented in Section 5.2 to models trained on different feature sets with limited memory.

Table 5.4: Testing accuracies (%) of game-based models trained and tested on the same game with consecutive time-windows (consecutive windows) compared to best models trained with a dynamic memory window (full history). Bold values show the best models for the given game and target output.

μA				
Test Game	**Consecutive Windows**			**Full History**
	Specific	**General**	**All**	
TinyCars	62.9±0.6	64.3±0.8	64.4±0.6	**64.8±1.2**
Solid	68.8±1.0	64.3±1.0	68.9±1.1	**73.2±0.7**
ApexSpeed	76.2±0.7	76±0.6	**76.8±0.7**	71.9±1.3
Heist!	73.8±0.7	73.7±0.8	79.0±0.8	**79.8±0.7**
TopDown	80.5±0.7	80.4±0.6	82.2±0.7	**83.5±1.1**
Shootout	58.7±1.4	59.0±1.1	58.5±1.3	**85.8±0.8**
Endless	73.6±1.1	72.4±1.2	**75.2±1.2**	69.5±1.8
Pirates!	66.2±0.5	64.4±0.5	65.8±0.4	**70.0±1.7**
Run'N'Gun	66.6±0.9	61.2±0.8	66.8±1.0	**79.8±1.9**

∇A				
Test Game	**Consecutive Windows**			**Full History**
	Specific	**General**	**All**	
TinyCars	53.5±0.5	53.6±0.5	53.5±0.4	**62.2±0.3**
Solid	57.7±0.4	55.1±0.4	58.1±0.3	**66.4±0.4**
ApexSpeed	63.3±0.5	62.2±0.4	63.9±0.4	**66.6±0.6**
Heist!	61.7±0.4	62.1±0.4	62.9±0.4	**70.3±0.3**
TopDown	64.8±0.7	66.1±0.4	67.1±0.7	**69.4±0.8**
Shootout	55.2±0.7	55.2±0.6	**55.9±0.8**	54.2±0.4
Endless	62.4±0.5	59.5±0.4	62.1±0.5	**72.0±1.0**
Pirates!	59.2±0.5	58.1±0.4	**59.4±0.4**	59.3±0.5
Run'N'Gun	59.7±0.5	60.9±0.5	61.7±0.5	**62.6±0.5**

Using the dynamic windowing method for memory generally increases model accuracy. In the case of μA models, in 4 out of 9 cases, using the session history significantly improves the best models trained on consecutive time-windows. The average increase in accuracy in these games is +12% on average. The most notable increases are Shootout with an average of +27% and Run'N'Gun with an average of +13%. In TinyCars, Heist!, and TopDown models, the performance is still improved, albeit not significantly. In these games, the improvement is only +0.8% on average. In ApexSpeed and Endless, the best μA models trained with consecutive time-windows are significantly better than the best models presented in Section 5.2. Here using a dynamic memory window leads to a -5% decrease in performance on average. The surprising improvement of μA Shootout models could be explained by the gameplay dynamics, which ramps up the intensity of the game over time. Since the game has just a few gameplay mechanics apart from aiming and shooting, the arousal of the players correlates strongly with this perceived intensity. Similarly, in Run'N'Gun, the level is designed to throw more and more enemies at the player with little to no incentives to stray from a straight path, giving a more robust temporal structure to the game. These results indicate that models that consider the entire session history can capture this well-structured temporal dynamic better. This observation is somewhat undermined by the contrasting results on ApexSpeed and Endless. However, it is possible that because these games have players react to proximal dangers with a minimal indication of what comes next (Endless due to the random nature of items, and ApexSpeed due to the low visibility of the track ahead), the long-term temporal dynamics of the games is less important than the most recent episode in evaluating the experience.

In the case of ∇A models, in 6 out of 9 games, using the full session history significantly improves the performance over the best models trained with consecutive time-windows. In these cases, the improvement is +6% on average, with the best improvement achieved on Endless (with +9%). From the remaining games, Run'N'Gun is improved only marginally, and both Shootout and Pirates! models perform significantly worse when trained on data using dynamic memory. In the latter cases, the decrease is −0.9% on average. It is interesting to note that while in the μA case, Endless models trained with a full session history performed significantly worse, in the case of ∇A they perform significantly better. This observation, coupled with the poor overall performance of ∇A Shootout models, point towards the ∇A output capturing the change in short-term play dynamics. Results presented here demonstrate that game-based models can generally benefit from increasing the memory window when applying *pairwise preference learning* to time-continuous data. The results make intuitive sense as an increased memory window would capture some of the time-related biases of the player experience.

5.5 Summary

This chapter detailed the results of game-based arousal modelling experiments, which establish a baseline for future chapters. In particular, it was found that μA is an easier task than ∇A, although algorithms performed relatively well in most cases (except for ∇A Shootout). The comparative results between genre-specific, general, and combined feature sets revealed that μA models trained of general features alone are quite robust and in the case of ∇A, genre-specific features still carry valuable information which improves model performance significantly in combined (genre-specific and general) feature sets. Section 5.3 revealed the non-linear importance of the most prominent features and Section 5.4 reexamined the results in light of the *memory* parameter. Results showed that μA models rely heavily on time-related features, while ∇A models focus more on genre-specific ones. While both outputs benefited from using dynamic time-windows, μA Shootout and Run'N'Gun models showed a surprisingly high-performance gain.

The next chapter focuses on genre-based modelling. Models in this chapter pool games together in the same genre and test the robustness of genre-specific and general features in terms of generality. Similarly to this chapter, the latter sections evaluate the importance of prominent features and the impact of the windowing technique on model performance.

Chapter 6

Genre-Based Arousal Modelling

A general application of AI and machine learning is a core interest to not just affective computing but computer science in general. However, in the field of emotion detection, many studies and datasets are still focused on *content-specific* modelling (see Section 2.2.2 for some examples), with fewer studies centred on personalised emotion recognition to create general user models (Snodgrass et al., 2019b). Studies on model generality in the field are often based on physiological (Martínez et al., 2011) or multi-modal features (Martínez and Yannakakis, 2014) both in the input-space of models and as possible markers for *affective content*. While these methods are often fruitful, they cannot be applied in real-world scenarios as they are costly and require researcher oversight. In the field of game research, the study towards general AI is predominantly focused on game-playing (Perez-Liebana et al., 2016a), with fewer user modelling applications (Togelius and Yannakakis, 2016; Yannakakis and Togelius, 2018). Even less attention is given to the prediction of *affective content*. Subsequently, *general affect modelling*—which aims to predict emotional outcomes of play in a game-independent way—is still in its infancy. The handful examples investigating this research avenue are often limited by ad-hoc testbeds and small datasets (Shaker et al., 2015; Shaker and Abou-Zleikha, 2016; Camilleri et al., 2017). Motivated by this lack of a comprehensive study on general affect modelling, this thesis leverages the *Affect Game AnnotatIoN* (AGAIN) dataset (see Chapter 4) to conduct a large-scale study of general affect modelling. To this end, first, this chapter focuses on *genre-based* general affect modelling.

Similarly to Chapter 5, for each game, three different feature sets are examined. A genre-specific, a general, and a combined set of the previous two. In addition, for every game and feature set, two different models are built. In the first case, models are using the *mean* of each annotation window to model the change in arousal (μA). In the second case, models are using the *average gradient* of each annotation window to model changes in the dynamics of the play experience (∇A); that is, the change in the acceleration of the arousal signal. The chapter is structured as follows: Section 6.1 focuses on the results of genre-based modelling. This section is divided between models aimed at unseen games within a genre (Section 6.1.1) and models aimed at unseen players of seen games within a genre (Section 6.1.2) to provide a complete picture of the use cases and robustness of genre-based modelling. Models use the same validation method and parameters as the baseline models introduced in Section 5.1. This means that all models are evaluated on *unseen players*. Additionally, in Section 6.2 RF feature importance is observed and in Section 6.3 the results are reexamined in light of their sensitivity to the windowing method (i.e. consecutive time-windows and dynamic memory windows). Finally, Section 6.4 summarises the results of

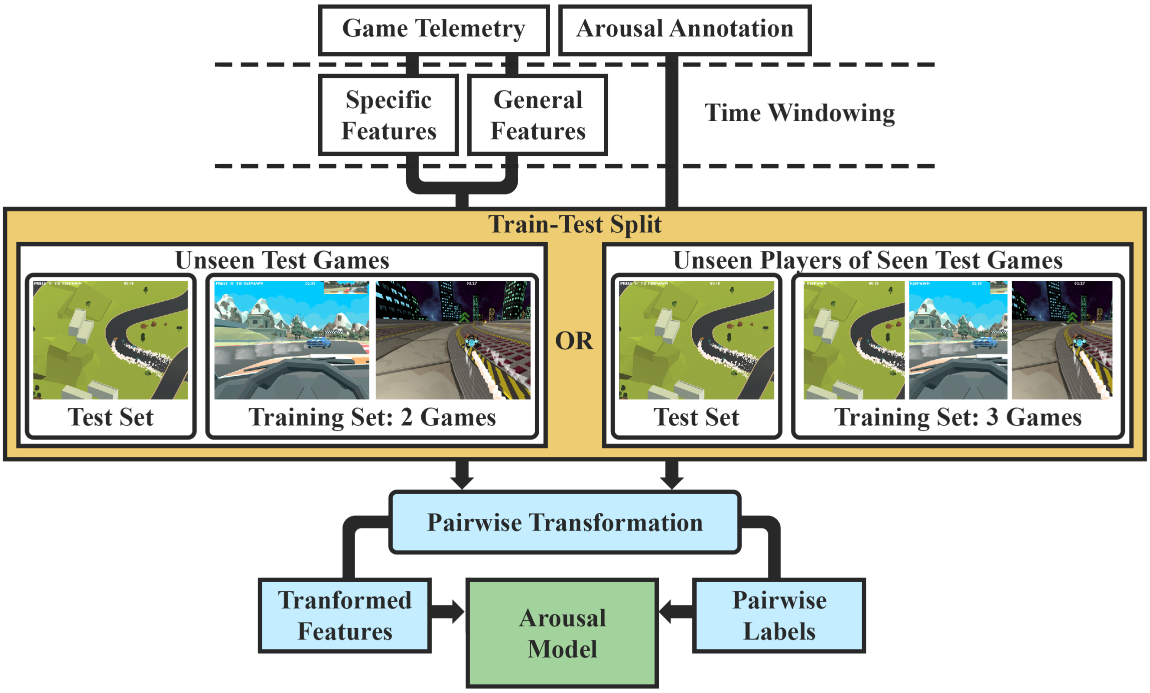
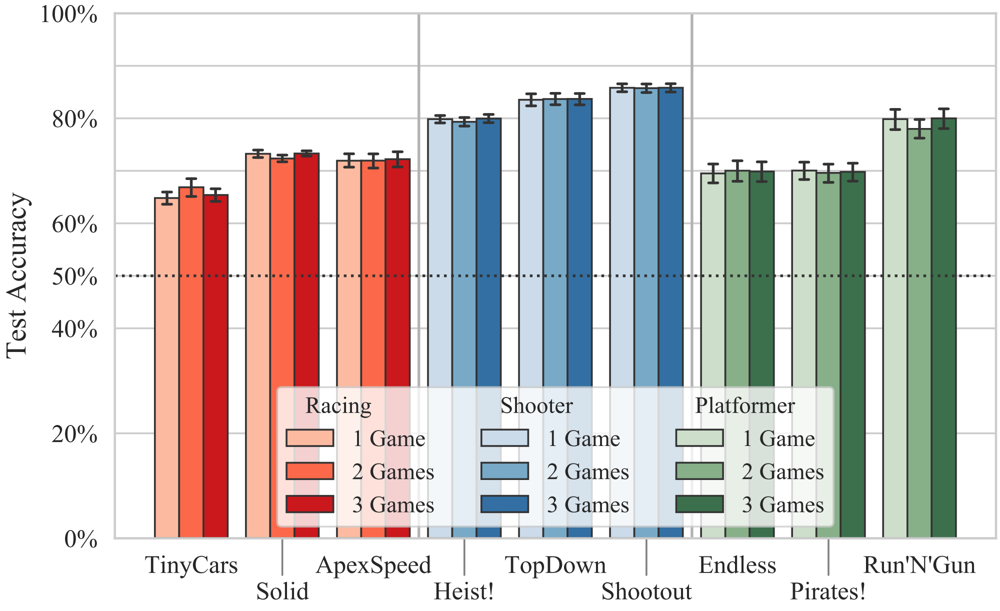


Figure 6.1: Genre-based modelling pipeline for modelling arousal across different unseen games of the same genre. Both genre-specific and general features are extracted from the raw telemetry. Models for unseen games are trained on data from two games and tested on an unseen game within the genre. Models for seen games are trained on all three games within a genre. Preference learning is applied by using a Pairwise Transformation, in which the ranking problem is reformulated as binary classification of pairwise labels (see Section 2.3 for more details).

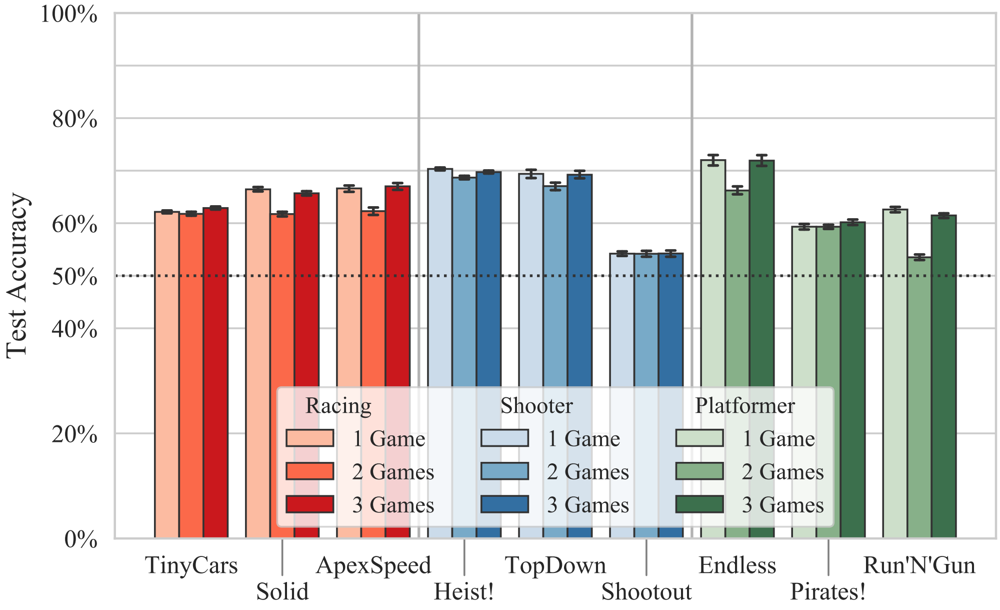
this Chapter.

6.1 Genre-Based Model Performance

This section discusses the main results of genre-based modelling. Models are referred to based on their test game. Models in Section 6.1.1 are built based on 2 games within the same genre and tested on the unseen one (leaving one game out). For example, the model for TinyCars is trained on Solid and ApexSpeed and tested on TinyCars. These experiments focus on the generality of features when it comes to modelling unknown games. Models in Section 6.1.2 are trained on all 3 games within a genre and predict arousal of unknown players in seen games. For example, the model for TinyCars is trained on TinyCars, Solid, and ApexSpeed and tested on unseen players of TinyCars. The purpose of this investigation is to see the robustness of general features compared to genre-specific ones and the effect of additional genre-level information on these models. Figure 6.1 illustrates both pipelines. Figure 6.2 gives an overview of the best models discussed in this chapter, while Tables 6.1 and 6.2 provide a more detailed report of the results.



(a) Models trained with μ_A arousal.



(b) Models trained with ∇A arousal.

Figure 6.2: Overview of the performance of genre-based models trained on the best feature set for each game in the dataset. Models are trained on either 2 or 3 games. Models trained on 2 games are tested on the unseen game (see Section 6.1.1); models trained on 3 games are tested on games within their genre (see Section 6.1.2). The hue shows the number of training games. The best models from Chapter 5 are included for comparison (1 Game models). The dotted line shows the natural baseline and the error bars indicate a 95% confidence interval.

Table 6.1: Testing accuracies (%) of models trained on 2 games and tested on an unseen game in the same genre compared to the best game-based models of Chapter 5. Bold values show the best models for the given game and target output.

μA				
Test Game	**Genre models**			**Game models**
	Specific	**General**	**All**	
TinyCars	66.3±1.6	66.2±1.5	**66.9±1.7**	64.8±1.2
Solid	70.6±0.6	72.2±0.5	72.3±0.6	**73.2±0.7**
ApexSpeed	67.2±1.0	**71.9±1.4**	69.9±1.2	71.9±1.3
Heist!	64.2±0.5	79.3±0.9	79.2±0.9	**79.8±0.7**
TopDown	76.3±1.0	83.5±1.1	**83.7±1.1**	83.5±1.1
Shootout	74.3±0.6	85.8±0.8	85.5±0.8	**85.8±0.8**
Endless	67.4±1.4	**70.0±2.0**	69.8±1.8	69.5±1.8
Pirates!	66.2±1.2	69.5±1.7	69.6±1.7	**70.0±1.7**
Run'N'Gun	62.1±0.9	74.6±1.4	78.0±1.7	**79.8±1.9**

∇A				
Test Game	**Genre models**			**Game models**
	Specific	**General**	**All**	
TinyCars	61.8±0.4	60.6±0.4	61.3±0.4	**62.2±0.3**
Solid	61.7±0.4	60.6±0.5	61.7±0.4	**66.4±0.4**
ApexSpeed	61.6±0.6	62.3±0.7	62.3±0.7	**66.6±0.6**
Heist!	68.0±0.3	67.8±0.4	68.7±0.3	**70.3±0.3**
TopDown	67.1±0.7	62.2±0.7	65.4±0.7	**69.4±0.8**
Shootout	**54.2±0.6**	50.6±0.4	53.1±0.5	54.2±0.4
Endless	66.2±0.8	62.3±0.6	65.1±0.7	**72.0±1.0**
Pirates!	57.9±0.4	56.5±0.4	59.3±0.4	**59.3±0.5**
Run'N'Gun	51.0±0.4	53.5±0.5	52.0±0.4	**62.6±0.5**

6.1.1 Modelling Unseen Games

Table 6.1 shows the performance of genre models for unseen games in terms of accuracy. The results of μA models reveal the robustness of general features in comparison to genre-specific ones. In 6 out of the 9 games tested (all except TinyCars, Endless, and Pirates!), μA models trained on genre-specific features perform significantly worse than ones trained on feature sets containing general features. In these cases, models trained on the specific features have an average of -8% drop in performance. The effect is most prominent in games that feature enemy projectiles and some form of shooting mechanics (-11% on average). Interestingly, Run'N'Gun—while it includes shooting and projectiles—does not feature mouse controls, suggesting that the performance difference between genre-specific and general models is not due to the different control scheme but the shooter-like gameplay dynamics. The difference in racing games is more marginal (-1% on average) but still significant. While in 5 out of 9 cases, game-based μA models perform better than genre-based models; there is no significant difference between the best genre and best game models. There is no significant difference between models trained on general, genre-specific and combined features. Furthermore, there is no significant difference between these models and the best game models of Section 5.2 (also included on Table 6.1). The average performance of the best genre models is the same as the game-specific models (70%, 83%, and 73% for racing, shooter, and platformer games, respectively). Interestingly, some genre models perform better than game models trained and tested on a single game. This is the case for TinyCars, ApexSpeed, TopDown, and Endless. However, this improvement is marginal (+1% on average).

Similarly to game models, ∇A genre-based models trained on genre-specific features show more robustness than models trained on general features. More specifically, models trained on genre-specific features are significantly better in 7 out of 9 cases (except for ApexSpeed, and Run'N'Gun) when compared to models trained on general features. Moreover, models trained on Heist! and Shootout and tested on TopDown perform better on genre-specific features than general features and a combined set of both genre-specific and general features. In the cases mentioned above, models trained on genre-specific features show an average improvement of +2%. The average performance of the ∇A models is 62%, 63%, and 60% for racing, shooter, and platformer games, respectively. Except for TinyCars, Shootout, and Pirates!, game-based models perform significantly better than genre-based ones. This result makes sense in light of previous experiments presented in Chapter 5, as ∇A models showed a higher sensitivity to the specificities of singular games.

6.1.2 Modelling Unseen Players of Seen Games

The most apparent application of genre-based modelling is to predict the outcomes of unseen games within the same genre. The intuitive expectation here is that even if we cannot make predictions at the same level of accuracy as with specialised game-based models, there is great utility in making snap predictions about unseen games. However, results presented in Section 6.1.1 show that in some cases, genre-based models can be as (or even more) robust as game-based models. Therefore it is worth investigating whether genre-based models which include a set of games from the same genre, including the target game, can improve the prediction performance.

Table 6.2 shows the comparative performance of these models and models from previous sections in terms of accuracy. Genre-based μA models for seen games show marginal improvement over the corresponding game-based models in 7 out of the 9 cases. These models unsurprisingly also improve on the genre-based models for unseen games in 6 out of the 9 cases. However, the only significant increase is in the case of models tested on Run'N'Gun (+2% accuracy on average). Interestingly, in the case of TinyCars and Endless, the inclusion of the target game in the training set is detrimental to the model performance. In the former case, this effect is a significant decrease of −2%.

In contrast, ∇A models only show marginal improvement over the game-based models in 3 out of the 9 cases. This improvement is only significant in the case of TinyCars (+1% accuracy on average). In contrast to using μA models, where some games were predicted better by genre-based models trained on two games, similar ∇A models are not as successful. Except for Pirates! and Shootout, models trained on three games are significantly better than models trained on two games. Due to the established reliance on genre-specific features and game-specific dynamics, ∇A genre-based models are improved when the target game is added to the training set with an average of +3%.

6.2 Impact of Features on Model Performance

Just as in Chapter 5 for game models, this section investigates the feature importance of genre models to gain a better understanding of how the models operate on their own and

Table 6.2: Testing accuracies (%) of models trained on 3 games and tested on an seen game in the same genre compared to the best genre-based models trained on two games from Section 6.1.1 and best game-based models from Chapter 5. Bold values show the best models for the given game and target output.

μA					
Test Game	**Genre models**				**Game models**
	3 games			**2 games**	**1 game**
	Specific	**General**	**All**		
TinyCars	64.2±1.4	65.4±1.3	64.8±1.3	**66.9±1.7**	64.8±1.2
Solid	73.1±0.5	73.0±0.5	**73.3±0.5**	72.3±0.6	73.2±0.7
ApexSpeed	70.8±1.3	**72.2±1.3**	71.2±1.2	71.9±1.4	71.9±1.3
Heist!	79.1±0.8	79.7±0.8	**80.0±0.8**	79.3±0.9	79.8±0.7
TopDown	82.3±1.1	83.5±1.1	83.7±1.1	**83.7±1.1**	83.5±1.1
Shootout	85.8±0.8	85.8±0.8	**85.8±0.8**	85.8±0.8	85.8±0.8
Endless	69.7±1.9	69.9±2.1	69.6±1.8	**70.0±2.0**	69.5±1.8
Pirates!	69.7±1.6	69.4±1.7	69.8±1.7	69.6±1.7	**70.0±1.7**
Run'N'Gun	79.6±1.9	80.0±1.9	**80.0±1.9**	78.0±1.7	79.8±1.9

μA					
Test Game	**Genre models**				**Game models**
	3 games			**2 games**	**1 game**
	Specific	**General**	**All**		
TinyCars	**62.9±0.3**	60.5±0.3	62.0±0.3	61.8±0.4	62.2±0.3
Solid	64.8±0.5	62.3±0.5	65.7±0.4	61.7±0.4	**66.4±0.4**
ApexSpeed	**67.0±0.6**	65.7±0.7	66.6±0.6	62.3±0.7	66.6±0.6
Heist!	69.1±0.3	69.6±0.4	69.7±0.3	68.7±0.3	**70.3±0.3**
TopDown	69.0±0.8	67.1±0.7	69.2±0.7	67.1±0.7	**69.4±0.8**
Shootout	54.1±0.5	52.1±0.5	54.2±0.6	54.2±0.6	**54.2±0.4**
Endless	71.8±1.1	68.7±0.8	71.9±1.1	66.2±0.8	**72.0±1.0**
Pirates!	59.7±0.5	59.1±0.5	**60.2±0.5**	59.3±0.4	59.3±0.5
Run'N'Gun	59.5±0.5	60±0.5	61.5±0.5	53.5±0.5	**62.6±0.5**

in comparison to game-based models. Tables 6.3 and 6.4 show the top five features for genre-based models built to predict unseen and seen games respectively. For a detailed breakdown of all features across genre-based models, see the tables in Appendix A.3.2 for models built to predict unseen and Appendix A.3.3 for models built to predict seen games. With μA models, a familiar pattern emerges. Time Passed and Player Score remain—unsurprisingly—the most prominent features in these models. Comparing game models to the two genre models, we can see a steady shift in weights towards these top two features. While models trained on one game had an average of 0.11 MDI score for these features, models trained on two games have 0.15, and models trained on three games have 0.17. This observation points towards less prominent features being filtered out as the variance of the input space increases. Interestingly—as it is also evidenced by Table 6.2—this shift does not have a clear diminishing return in μA models. This filtering effect also present in ∇A models. The top features gain prominence as the input variance increases with new games added to the training set. While there was a more substantial shift in weights between

Table 6.3: Feature importance of genre models trained on 2 games, as derived from the Random Forests, averaged across games of the same genre. Features are labelled as general (G) or specific (S). Features present in top five features of all models with the same output are shown in bold.

Genre	μA			∇A		
		Feature	**Score**		**Feature**	**Score**
Racing	G	**Time Passed**	0.116	G	Time Passed	0.046
	G	**Player Score**	0.110	S	Player Collision	0.043
	S	Player GasPedal	0.066	G	Player Score	0.042
	G	Player Activity	0.038	S	Player Speed	0.038
	S	Bot Collision	0.038	G	Player Movement	0.033
Shooter	G	**Time Passed**	0.225	S	Bot Shooting	0.064
	G	**Player Score**	0.162	G	Bot Diversity	0.052
	S	Bot Reloading	0.042	S	Bot Proj. Count	0.049
	S	Player Health	0.040	S	Bot Health	0.048
	G	Bot Diversity	0.037	S	Player Damaged	0.043
Platformer	G	**Time Passed**	0.137	G	Bot Movement	0.053
	G	**Player Score**	0.132	S	Player Health	0.042
	S	Player Damaged	0.046	G	Bot Count	0.042
	S	Player Death	0.035	G	Bot Diversity	0.033
	G	Bot Movement	0.032	S	Bot SpeedX	0.030

Table 6.4: Feature importance of genre models trained on 3 games, as derived from the Random Forests, averaged across games of the same genre. Features are labelled as general (G) or specific (S). Features present in top five features of all models with the same output are shown in bold.

Genre	μA			∇A		
		Feature	**Score**		**Feature**	**Score**
Racing	G	**Time Passed**	0.136	G	Time Passed	0.050
	G	**Player Score**	0.132	S	Player Collision	0.049
	S	Player Gas Pedal	0.075	G	Player Score	0.046
	S	Bot Collision	0.040	S	Player Speed	0.041
	S	Player Lap	0.034	S	Player Speed Boost	0.038
Shooter	G	**Time Passed**	0.259	S	Bot Shooting	0.069
	G	**Player Score**	0.183	G	Bot Diversity	0.054
	S	Bot Reloading	0.052	S	Bot Proj. Count	0.054
	S	Player Health	0.037	S	Bot Proj. Player Dist.	0.052
	G	Bot Diversity	0.037	S	Bot Health	0.049
Platformer	G	**Time Passed**	0.163	G	Bot Movement	0.060
	G	**Player Score**	0.149	G	Bot Count	0.048
	S	Player Damaged	0.058	S	Player Health	0.046
	S	Player Death	0.039	G	Bot Diversity	0.037
	S	Player Collision Left	0.029	S	Bot Health	0.031

Table 6.5: Testing accuracies (%) of genre models trained on 2 games and tested on an unseen game with consecutive time-windows (consecutive windows) compared to best models trained with a dynamic memory window (full history). Bold values show the best models for the given game and target output.

μA				
Test Game	**Consecutive Windows**			**Full History**
	Specific	**General**	**All**	
TinyCars	59.9±0.8	59.3±0.9	60.5±0.9	**66.9±1.7**
Solid	61.5±0.8	56.1±0.5	60.9±0.5	**72.3±0.6**
ApexSpeed	64.5±0.9	67.0±1.0	67.2±0.9	**71.9±1.4**
Heist!	72.3±0.9	67.5±0.7	75.2±0.8	**79.3±0.9**
TopDown	80.4±0.7	75.9±0.5	82.7±0.6	**83.7±1.1**
Shootout	58.2±1.5	54.1±1.1	57.7±1.1	**85.8±0.8**
Endless	63.0±0.7	51.7±0.9	60.8±0.8	**70.0±2.0**
Pirates!	62.9±0.5	57.5±0.9	62.1±0.5	**69.6±1.7**
Run'N'Gun	56.3±0.7	54.2±0.6	58.0±0.7	**78.0±1.7**

∇A				
Test Game	**Consecutive Windows**			**Full History**
	Specific	**General**	**All**	
TinyCars	54.4±0.5	53.3±0.5	54.1±0.4	**61.8±0.4**
Solid	57.4±0.4	53.6±0.4	56.1±0.3	**61.7±0.4**
ApexSpeed	51.3±0.3	53.1±0.4	53±0.4	**62.3±0.7**
Heist!	61.1±0.4	61.9±0.5	63.4±0.5	**68.7±0.3**
TopDown	63.8±0.5	62.6±0.6	65±0.5	**67.1±0.7**
Shootout	55±0.8	54.9±0.5	**55.9±0.9**	54.2±0.6
Endless	59.8±0.5	57±0.3	59±0.5	**66.2±0.8**
Pirates!	58.2±0.5	56.6±0.5	58.6±0.6	**59.3±0.4**
Run'N'Gun	55±0.3	54.5±0.4	**55.5±0.3**	53.5±0.5

models trained one and two games, adding a third game to the training set does not disrupt the ranked order of the top features as much. In this regard, the most interesting results are the shift away from Player Damaged in shooter game models and from horizontal speed (both Bot and Player Speed X) in platformer game models.

6.3 Sensitivity to Memory

This section reexamines the results of genre-based models trained on 2 and 3 games in the context of the windowing method used during the pairwise transformation. It has been observed in Chapter 5 that expanding the memory window during the pairwise transformation of the dataset to encompass the whole available session history generally increases the performance of the models. The same general pattern can be observed with genre models as well to an even greater degree. Table 6.5 shows the results of models trained on 2 games to predict arousal in unseen games, while Table 6.6 shows the results of models trained on 3 games to predict arousal in seen games.

The impact of dynamic memory windowing is the highest in genre-based μA models

Table 6.6: Testing accuracies (%) of genre models trained on 3 games and tested on an unseen game with consecutive time-windows (consecutive windows) compared to best models trained with a dynamic memory window (full history). Bold values show the best models for the given game and target output.

μA				
Test Game	**Consecutive Windows**			**Full History**
	Specific	**General**	**All**	
TinyCars	62.2±0.8	64.5±1.1	64.2±0.7	**64.8±1.3**
Solid	68.7±0.9	63.1±0.8	66.4±0.9	**73.3±0.5**
ApexSpeed	75.3±0.8	75±0.8	**76.0±0.7**	72.2±1.3
Heist!	73.5±0.7	72.6±0.7	78.6±0.9	**80.0±0.8**
TopDown	80.9±0.7	80.1±0.6	82.7±0.7	**83.7±1.1**
Shootout	60.4±1.5	58.2±1.5	60.6±1.6	**85.8±0.8**
Endless	70.6±1.0	67.4±1.1	**73.0±1.1**	69.9±2.1
Pirates!	66.3±0.6	62.7±0.6	66.8±0.4	**69.8±1.7**
Run'N'Gun	66.4±0.8	57.3±0.6	66.8±0.8	**80.0±1.9**

∇A				
Test Game	**Consecutive Windows**			**Full History**
	Specific	**General**	**All**	
TinyCars	55.0±0.6	53.6±0.6	54.8±0.5	**62.9±0.3**
Solid	58.6±0.4	55.6±0.3	58.1±0.3	**65.7±0.4**
ApexSpeed	63.3±0.5	61.5±0.3	63.7±0.4	**67.0±0.6**
Heist!	62.0±0.4	62.5±0.5	63.7±0.5	**69.7±0.3**
TopDown	65.1±0.6	65.7±0.5	67.0±0.5	**69.2±0.7**
Shootout	55.1±0.8	55.5±0.6	**55.9±0.8**	54.2±0.6
Endless	61.9±0.4	58.4±0.3	62.0±0.4	**71.9±1.1**
Pirates!	58.8±0.5	58.3±0.5	59.4±0.5	**60.2±0.5**
Run'N'Gun	59.4±0.4	60.1±0.5	61.4±0.5	**61.5±0.5**

trained on 2 games to predict unseen games. Not only the models considering the entire session history have higher performance than models only considering consecutive time-windows, but this increase is also significant in almost all cases (except for TopDown). The average increase in performance is +10%, with Shootout and Run'N'Gun showing an average improvement of +28% and +20% respectively. The impact of dynamic memory windowing is similar to the case of game-based models in genre-based ∇A models trained on 2 games. Using a dynamic session history improves 6 out of the 9 models with an average of +6%. The performance tested on Shootout and Run'N'Gun is lower by an average −2%, and the difference is not significant in the case of Pirates!.

In contrast to genre-based models trained on 2 games, the weight of dynamic memory windowing is lower in genre models trained on 3 games to predict seen games. Genre-based μA model performance has improved significantly only in 4 out of 9 cases (by an average of +12%). The performance of the racing genre and platformer genre models tested on ApexSpeed and Endless respectively decrease by an average of −3.5% when dynamic memory windows are applied. Additionally, racing genre models tested on TinyCars and shooter genre models tested on Heist! and TopDown do not show a significant improvement.

Genre-based ∇A models trained on 3 games also show an average improvement of +6% with a dynamic memory windowing in 6 out of 9 cases. The only game where the model performance decreases is Shootout (by −2%). Platformer genre models tested on Pirates! and Run'N'Gun do not improve significantly.

These results show that genre-based preference learning models trained on multiple games can benefit from an increased memory even more so than game-based models introduced in Chapter 5. This is especially true for models trained on 2 games to predict arousal in unseen games. The effect can be explained by the smoothing effect of the large memory window, which could decrease the noise of the dataset. As the variance of training sets including multiple games is higher compared to training sets including only a single game, this de-noising is essential for increasing model performance above a 70% average accuracy in most cases.

6.4 Summary

This chapter presented a robust approach to genre-based arousal modelling. Experiments in the chapter focused on two use-cases. First, genre-based modelling of unseen games, in which models are trained on two games and tested on an unseen one. Second, genre-based modelling of seen games, in which models are trained on three games within a genre and tested on a game also included in the training set. Models were compared to a baseline of game-based models trained and tested on the same game and previously presented in Chapter 5. Results show that μA genre models could not only match the performance of game-based models but, in many cases, surpass them. This is especially true for models predicting seen games. Feature importance in these models is shifted towards time-related game-agnostic features. This weighting of features combined with a smoothing effect of an extended memory window during the data processing leads to a surprising level of robustness (average performance of 70%, 83%, and 73% for racing, shooter, and platformer games, respectively). On the other hand, due to the reliance on genre-specific features, ∇A models show worse performance when predicting unseen games and comparable performance to game models when predicting seen games. In the former case, the average model performance reaches 62%, 63%, and 60% and in the latter case 65%, 64%, and 65% for the racing, shooter, and platformer games, respectively. However, as the ∇A output has no strong correlation with time-related features, which contribute to the robustness of μA models, these results make intuitive sense.

The next chapter concludes the investigation of general affect modelling by examining genre- and game-agnostic models. The first half of this chapter reexamines models from Chapter 5 and 6 in terms of how well they can be transferred to previously unseen games. The latter half of the chapter focuses on models composed of games from different genres. This chapter also concludes the investigation of feature importance in general models and the impact of dynamic memory windows on model performance.

Chapter 7

General Arousal Modelling

In previous chapters, models have been created based on data from single games and games from the same genre. Even though analysis of RF model feature importance showed that game-agnostic features are highly relevant to the modelling of arousal (especially in case of modelling the change in the level of arousal—μA), the best performance was often attributed to models which included genre-specific features. As the next step towards general affect modelling, this chapter looks at truly general arousal models that rely solely on high-level game-agnostic features to create models which can predict arousal in any game of any genre. To test the feasibility of this method, general models that are presented here use hand-crafted heuristic features (see also Section 4.3.4). The goal of these general features is to create models which can be used to make snap predictions of completely unseen games. While the industry application of such models is apparent, the study of affective computing and artificial intelligence, in general, can benefit from such models as a step towards domain-independent general models of affect and social intelligence (Togelius and Yannakakis, 2016). As mentioned before in Chapter 2, studies have been done before on games from dissimilar genres; however, as evidenced by results presented in Chapter 6, differences in individual games used for training can lead to stark differences in model performance. This chapter addresses this limitation by using a more methodical approach and a more complex dataset than previous studies in the field. To get a more comprehensive picture of how robust heuristic general features are, experiments in this chapter focus on building models based on multiple games and testing models on unseen ones. This chapter examines general affect modelling across different genres through two different use-cases. First, general modelling of unseen genres, in which models are trained on 6 games from two genres and tested on the games of the unseen genre. Second, general modelling of unseen games, in which models are trained on 8 games and tested on the unseen game. The former models show the robustness of the method by modelling games from unknown genres. The latter models show the upper boundary of the performance of the method in terms of accuracy by maximising the amount of information in the input space. All models presented in this section—including genre- and game-based models presented in comparison to the general models—are trained on general features (see Section 4.3.4). Just as in previous chapters, models presented here use the same parameters and validation method as presented in Section 5.1.

The chapter is divided into six main sections. Section 7.1 and 7.2 reexamine models constructed in previous chapters from the perspective of general modelling and look into their performance across all games (regardless of genre). Section 7.3 presents the results of general models trained on both 6 and 8 games compared to models from previous chapters.

Feature importance of general models are observed in Section 7.4. Section 7.5 reexamines the new results presented in this chapter in light of dynamic memory windowing to round out the investigation started in Chapter 5. Finally, Section 7.6 gives a summary of the Chapter.

7.1 Generality of Game-Based Models

As a preliminary step towards general modelling, this section investigates the robustness of game-based models in game-agnostic arousal prediction. In this section, models are trained on one game and tested on all games in the dataset. The section is structured by the genre of the training games. Because models in this section predict games outside of their genre, all models presented here are trained on general features only. Tables 7.1, 7.2, and 7.3 show the results of these experiments. While it is expected that game-based models perform better on the games they were trained on, results presented here show that this is not necessarily the case.

7.1.1 Racing Games

Table 7.1 presents the performance of racing game models in terms of accuracy across all games. Racing game μA models surprisingly perform significantly better when tested on a different game than what they were trained on in many cases. In 7 out of 8 cases, models trained on TinyCars, in 4 out of 8 cases models trained on Solid, and 5 out of 8 cases models trained on ApexSpeed perform better on games other than what they were trained on. On average racing game models have a 69% accuracy on racing games, 80% accuracy on shooter games, and 71% accuracy on platformer games. Most interestingly, all racing game models perform exceptionally well in shooter games. Overall there is a +11% improvement when these games are tested on shooter instead of racing games. As subsequent sections will also show, this robustness in predicting shooter games is true to most models. Interestingly, the same effect can also be observed on Run'N'Gun (+7% on average compared to racing games), which is a game also featuring projectiles, suggesting that the observed effect is due to the play dynamics created by shooting mechanics rather than the control scheme unique to shooter games (see a similar observation also in Section 6.1.1). While on average models trained on Solid and ApexSpeed (75% and 74% respectively) perform similarly, models trained on TinyCars are the weakest across the board (71% on average)—even though most improvement can be seen on TinyCars models when tested on other games. This suggests that TinyCars is an especially hard game to predict compared to others.

Racing game ∇A models unsurprisingly perform the best when tested on other racing games. Models trained on these games have an average of 61% accuracy on racing games, 55% accuracy on shooter games, 54% accuracy on platformer games. The most robust of ∇A models is ApexSpeed with an overall average performance of 59% followed by Solid with 56% and TinyCars with 55%. These results again show the difficulty of TinyCars compared to other racing games.

7.1.2 Shooter Games

Table 7.2 presents the performance of shooter game models in terms of accuracy across all games. Shooter game μA models mostly perform better on their own games as expected, with a few exceptions. Models trained on Heist! perform significantly better on

Table 7.1: Testing accuracies (%) of racing game-based models across different games in the dataset. Columns show the training games and rows show the testing games. Bold values show results where models are performing significantly better than on the games they were trained on. Underlined values show results where models are trained and tested on the same game.

μA			
Test Game	**TinyCars**	**Solid**	**ApexSpeed**
TinyCars	64.3±0.9	65.8±1.4	66.2±1.1
Solid	**67.4±0.6**	73.2±0.7	**73.5±0.6**
ApexSpeed	**67.7±1.2**	72.3±1.3	71.9±1.3
Heist!	**75.6±0.7**	**78.2±0.7**	**78.3±0.9**
TopDown	**76.4±0.8**	**80.5±1.0**	**81.4±1.0**
Shootout	**81.0±1.0**	**84.3±0.9**	**84.7±0.7**
Endless	**67.3±1.5**	69.7±1.8	69.0±2.0
Pirates!	65.5±1.3	68.7±1.7	67.9±1.6
Run'N'Gun	**71.9±1.3**	**78.7±1.9**	**77.1±1.8**

∇A			
Test Game	**TinyCars**	**Solid**	**ApexSpeed**
TinyCars	60.2±0.3	60.6±0.4	57.7±0.4
Solid	59.9±0.3	63.6±0.4	58.7±0.5
ApexSpeed	60.6±0.6	62.5±0.7	65.7±0.6
Heist!	51.1±0.4	57.2±0.3	66.3±0.4
TopDown	54.4±0.3	54.8±0.5	60.7±0.5
Shootout	51.1±0.5	51.2±0.5	50.8±0.4
Endless	53.1±0.5	54.1±0.4	65.0±0.5
Pirates!	53.1±0.5	50.7±0.6	54.3±0.4
Run'N'Gun	53.1±0.4	51.3±0.3	54.7±0.6

the other two shooter games, and models trained on TopDown perform significantly better on Shootout than on TopDown. On average shooter game models have a 68% accuracy on racing games, 80% accuracy on shooter games, and 71% accuracy on platformer games. In terms of robustness, Heist! and TopDown performs similarly well across all games (76% on average) with models trained on Shootout doing significantly worse (68% on average). The weakness of Shootout models is interesting bacause across all μA models (including racing and platformer games as well) it is by far the easiest to predict. As Shootout has the least amount of game mechanics (only look around and shoot, while reloading is automatic), this result suggests that this game is too simple to capture more complex gameplay dynamics. Models trained both on Heist! and TopDown could predict μA arousal in Run'N'Gun with 80% accuracy. This result reinforces the hypotesised similarity in the play experience in 3D shooter games and 2D platformer-shooter hybrids.

On average, shooter game ∇A models have a 55% accuracy on racing games, 59% accuracy on shooter games, and 57% accuracy on platformer games. Models trained on Heist! and TopDown perform better on their own games than on other games. In contrast, in 6 out of 8 cases, models trained on Shootout perform better on other games. This shows the robustness of the temporal dynamics captured by ∇A as it can be learned from a

Table 7.2: Testing accuracies (%) of shooter game-based models across different games in the dataset. Columns show the training games, and rows show the testing games. Bold values show results where models are performing significantly better than on the games they were trained on. Underlined values show results where models are trained and tested on the same game.

μA			
Test Game	**Heist!**	**TopDown**	**Shootout**
TinyCars	66.6±1.7	66.7±1.7	61.1±0.9
Solid	74.0±0.7	74.3±0.8	66.4±0.8
ApexSpeed	72.5±1.5	72.7±1.5	62.0±0.8
Heist!	79.4±0.7	79.3±0.9	67.0±0.7
TopDown	**83.5±1.1**	83.3±1.1	71.1±0.8
Shootout	**85.0±0.9**	**85.8±0.8**	85.8±0.8
Endless	70.9±2.0	70.8±2.1	64.7±1.2
Pirates!	69.6±1.8	69.7±1.8	63.4±0.8
Run'N'Gun	80.2±1.9	80.2±1.9	73.7±1.4

∇A			
Test Game	**Heist!**	**TopDown**	**Shootout**
TinyCars	53.3±0.5	55.8±0.4	**54.6±0.5**
Solid	54.9±0.7	55.5±0.6	**54.3±0.4**
ApexSpeed	55.8±0.6	59±0.5	**54.8±0.4**
Heist!	69.5±0.3	66.8±0.4	**57.7±0.3**
TopDown	61.5±0.6	67.3±0.7	**58.2±0.4**
Shootout	50.3±0.4	50.9±0.3	52.5±0.3
Endless	65.2±0.7	61.7±0.5	51.6±0.4
Pirates!	57.7±0.3	56.6±0.3	52.0±0.5
Run'N'Gun	54.7±0.5	56.6±0.4	**54.7±0.3**

relatively simple game such as Shootout. Additionally, ∇A models show less variance across the board compared to μA models in terms of performance. Interestingly, while Shootout is the easiest game to predict for μA models, ∇A models struggle with this simple game. An explanation for this phenomenon could be that the relative change in the level of arousal is more important in Shootout than the rate of change encoded by ∇A. In terms of the overall robustness, models trained on TopDown lead the list with an average performance of 59%, followed by Heist! with 58%, and finally Shootout with 54%.

7.1.3 Platformer Games

Table 7.3 presents the performance of platformer game models in terms of accuracy across all games. Similarly to racing game models, some platformer game μA models also perform significantly better on some shooter games than on platformer games, although to a lesser extent. The overall performance increase between models trained and tested on platformer games compared to ones tested on shooter games is +6% on average. On average, platformer game μA models have a 68% accuracy on racing games, 80% accuracy on shooter games, and 71% accuracy on platformer games. Models trained on Endless, Pirates!, and Run'N'Gun have an average performance of 69%, 71%, and 72% respectively across all

Table 7.3: Testing accuracies (%) of platformer game-based models across different games in the dataset. Columns show the training games and rows show the testing games. Bold values show results where models are performing significantly better than on the games they were trained on. Underlined values show results where models are trained and tested on the same game.

μA			
Test Game	**Endless**	**Pirates!**	**Run'N'Gun**
TinyCars	65.6±1.5	64.6±1.0	64.7±0.9
Solid	71.2±0.9	62.9±0.6	68.5±0.7
ApexSpeed	65.8±1.1	66.8±1.2	67.1±1.4
Heist!	**72.0±0.6**	**75.2±0.9**	70.6±0.6
TopDown	**73.9±1.0**	**75.3±1.1**	77.0±1.0
Shootout	70.7±1.0	**83.9±0.9**	**84.4±0.9**
Endless	69.1±1.8	69.1±1.8	68.5±1.5
Pirates!	63.5±1.4	68.9±1.7	69.3±1.7
Run'N'Gun	68.7±1.5	69.4±1.0	79.8±1.9

∇A			
Test Game	**Endless**	**Pirates!**	**Run'N'Gun**
TinyCars	55.8±0.5	53.7±0.4	56.3±0.4
Solid	56.1±0.3	54.8±0.7	56.6±0.5
ApexSpeed	61.5±0.4	56.2±0.6	60.5±0.6
Heist!	61.1±0.5	**67.5±0.5**	**63.7±0.4**
TopDown	57.5±0.3	59.4±0.5	**63.7±0.6**
Shootout	50.0±0.3	51.0±0.5	52.0±0.5
Endless	71.1±0.9	62.9±0.6	58.6±0.4
Pirates!	53.8±0.2	59.1±0.5	55.2±0.6
Run'N'Gun	51.0±0.4	56.1±0.5	61.8±0.5

games. Interestingly, despite the surprisingly high performance of shooter games predicting Run'N'Gun, models trained on Run'N'Gun do not match the performance of models trained and tested on shooter games. It is possible that because Run'N'Gun is a 2D game, while the 3D shooters could capture its dynamics, data based on the 2D environment does not contain enough information to predict the gameplay experience of 3D games to the same degree.

Platformer game ∇A models trained on Pirates! interestingly predict Heist! with higher accuracy (+3% on average) than their own game. Similarly, models trained on Run'N'Gun predict both Heist! and TopDown with higher accuracy (+2% on average). While the connection between Run'N'Gun and shooter games has already been established, the connection between Heist! and Pirates! is not clear. While Pirates! is a relatively linear 2D game, Heist! has a fairly open-ended 3D map with very different gameplay mechanics. On average, platformer game ∇A models have a 57% accuracy on racing games, 58% accuracy on shooter games, and 59% accuracy on their own games. Models trained on Endless and Pirates! have an average of 58% accuracy, while models trained on Run'N'Gun have an average performance of 59% across all games.

Table 7.4: Testing accuracies (%) of racing genre-based models across different games in the dataset. Columns show the training games, and rows show the testing games. Models titled "Racing Games" include all three racing games. Bold values show results where models perform significantly better than on all of the games included in their training set. Underlined values show results where the test game was included in the model's training set.

μA				
Test Game	**TinyCars & Solid**	**TinyCars & ApexSpeed**	**Solid & ApexSpeed**	**Racing Games**
TinyCars	65.2±1.2	65.3±1.1	66.2±1.5	65.4±1.2
Solid	72.9±0.6	72.2±0.5	73.4±0.7	73.0±0.5
ApexSpeed	71.9±1.4	71.9±1.3	72.4±1.3	72.2±1.4
Heist!	**78.7±0.8**	**78.9±0.9**	**79.2±0.8**	**79.2±0.9**
TopDown	**81.2±0.9**	**82.5±1.0**	**82.5±1.0**	**82.8±1.0**
Shootout	**83.0±1.0**	**84.4±0.8**	**85.5±0.8**	**84.6±0.9**
Endless	70.2±1.9	70.1±2.0	70.3±2.0	70.6±2.0
Pirates!	69.0±1.7	68.6±1.6	69.3±1.8	69.3±1.7
Run'N'Gun	**79.4±1.8**	**77.9±1.8**	**79.9±1.9**	**79.9±1.9**

∇A				
Test Game	**TinyCars & Solid**	**TinyCars & ApexSpeed**	**Solid & ApexSpeed**	**Racing Games**
TinyCars	60.6±0.3	60.0±0.3	60.6±0.4	60.5±0.3
Solid	62.8±0.4	60.6±0.5	63.0±0.5	62.3±0.5
ApexSpeed	62.3±0.7	65.4±0.6	65.9±0.6	65.7±0.7
Heist!	54.4±0.3	60.3±0.4	62.0±0.3	59.0±0.3
TopDown	55.0±0.4	60.0±0.5	57.6±0.5	57.2±0.5
Shootout	51.3±0.5	51.6±0.5	51.3±0.4	51.0±0.5
Endless	53.9±0.4	63.2±0.5	63.4±0.5	61.9±0.4
Pirates!	51.6±0.6	53.8±0.4	52.5±0.5	52.7±0.5
Run'N'Gun	53.1±0.3	55.4±0.5	53.8±0.4	54.3±0.4

7.2 Generality of Genre-Based Models

This section reexamines genre-based models in the context of general modelling. Just as with game-based models in Section 7.1, this section looks at how well genre-based models generalise to other genres. As in the previous section, all models presented here are trained on general features only. Tables 7.4, 7.5, and 7.6 show the results of these experiments. After learning about the robustness of game-based models, it is expected that games which were easier to predict for all game-based models will be similarly easy to predict here as well.

7.2.1 Racing Games

Table 7.4 presents the performance of racing genre-based models in terms of accuracy across all games. Just as with game-based models, many genre-based μA models improve significantly on model performance compared to tests done on games that are included in the models' training set. As it has been observed before, shooter games and Run'N'Gun are the easiest to predict. The average performance of models is 70&, 82%, and 73% for the racing, shooter, and platformer games, respectively. Unlike game-based models, the difference between individual models' average performance is low. The average performance of all racing genre models is 75%, which is a +2% overall improvement over the average of game-based models.

Racing genre ∇A models also show a similar picture as game-based models in Section 7.1. These models are performing the best on games that were included in their training set. The average performance of models is 62& for racing games, and 56%, for shooter and platformer games. Interestingly, models trained on the combined set of TinyCars and Solid can predict ApexSpeed significantly better than TinyCars (+1% on average). However, the performance of the combined models is very close to models trained on Solid alone, which means that adding TinyCars to these models does not increase the robustness. Similarly, models trained on all racing games are significantly better at predicting Endless than TinyCars (+1% on average). Here, the culprit behind the improvement is ApexSpeed. However, compared to the game-based ApexSpeed model (69% accuracy on Endless), racing genre models perform worse, indicating that adding new games is not always beneficial and can potentially confuse the models. Looking at the average model performance, models trained on TinyCars and Solid have an average accuracy of 56%, while models trained on the other two games have 59%. Including all racing games in the models results in an average accuracy of 58%, again pointing towards TinyCars confusing the models. Nevertheless, overall the genre-based models show some improvement over game-based models (+1% on average across all games).

7.2.2 Shooter Games

Table 7.5 presents the performance of shooter genre models in terms of accuracy across all games. Unsurprisingly, as shooter games are relatively easy to predict, in almost all cases μA models are performing better on shooter games than games from other genres. Models trained on Heist! and TopDown perform significantly better on Shootout than on their respective training games. Models trained on Heist! and Shootout perform significantly better on TopDown than on Heist!. On average shooter game models have a 71% accuracy on racing games, 83% accuracy on shooter games, and 74% accuracy on platformer games. This is an overall +3% improvement over game-based models. Similarly to racing genre models, there is no variation between the performance of μA shooter game models either, all models are performing at an average of 76%.

On average, shooter game ∇A models have a 57% accuracy on racing games, 62% accuracy on shooter games, and 59% accuracy on platformer games. Just as with μA models, this is an overall +3% improvement over game-based models. Individual model performance is 60% on average. While models which include Shootout are doing marginally worse, there is only 1% difference between these models and models not trained on Shootout. However, it seems that including all shooter games in the models somewhat mitigates this issue. As Shootout has proven to be very hard to capture by the ∇A of the arousal trace, models have better performance on other games even if they were trained on Shootout.

Table 7.5: Testing accuracies (%) of shooter genre-based models across different games in the dataset. Columns show the training games and rows show the testing games. Models titled “Shooter Games” include all three racing games. Bold values show results where models perform significantly better than on all of the games included in their training set. Underlined values show results where the test game was included in the model's training set.

μA				
Test Game	**Heist! & TopDown**	**Heist! & Shootout**	**TopDown & Shootout**	**Shooter Games**
TinyCars	66.5±1.7	66.6±1.7	66.7±1.7	66.5±1.7
Solid	74.4±0.8	74.1±0.7	74.3±0.8	74.4±0.8
ApexSpeed	72.8±1.5	72.5±1.5	72.6±1.5	72.8±1.5
Heist!	79.5±0.8	79.5±0.7	79.3±0.9	79.7±0.8
TopDown	83.6±1.1	83.5±1.1	83.4±1.1	83.5±1.1
Shootout	**85.8±0.8**	85.8±0.8	85.8±0.8	85.8±0.8
Endless	70.8±2.1	70.8±2.1	70.7±2.1	70.8±2.1
Pirates!	69.7±1.8	69.6±1.7	69.7±1.8	69.6±1.7
Run'N'Gun	80.2±1.9	80.2±1.9	80.2±1.9	80.2±1.9

∇A				
Test Game	**Heist! & TopDown**	**Heist! & Shootout**	**TopDown & Shootout**	**Shooter Games**
TinyCars	56.7±0.4	54.4±0.4	56.2±0.4	56.7±0.3
Solid	56.4±0.6	55.4±0.7	55.8±0.6	56.6±0.7
ApexSpeed	59.7±0.5	57.5±0.5	60.5±0.5	60.5±0.5
Heist!	69.3±0.4	69.4±0.3	67.8±0.4	69.6±0.4
TopDown	67.3±0.7	62.2±0.7	66.7±0.7	67.1±0.7
Shootout	50.6±0.4	52.1±0.4	52.2±0.4	52.1±0.5
Endless	63.3±0.6	65.3±0.6	61.8±0.5	63.3±0.5
Pirates!	57.3±0.3	58.1±0.3	57.2±0.4	57.7±0.3
Run'N'Gun	56.2±0.4	56.0±0.4	57.1±0.5	56.6±0.4

7.2.3 Platformer Games

Table 7.6 presents the performance of platformer genre models in terms of accuracy across all games. Just as with game- and genre-based racing game models and game-based platformer models μA platformer genre models also perform better on some shooter games. On average platformer genre μA models have a 69% accuracy on racing games, 80% accuracy on shooter games, and 73% accuracy on platformer games. Similarly to racing genre models, the improvement over the overall performance of game-based platformer models is only +1%. Interestingly, while in other genres pooling the games together did not increase performance over the average accuracy of two-game models, there is a marginal gain with platformer games. While Endless and Pirates! perform at 73% on average, and the other two-game models perform at 74% on average, including all games in the models is increasing the overall performance to 75%. In most cases, including all games do not improve the performance on a single game significantly. However, in the case of TopDown, pooling all platformer games together in the training set improves the performance of the best two-game model (Pirates!

Table 7.6: Testing accuracies (%) of platformer genre-based models across different games in the dataset. Columns show the training games, and rows show the testing games. Models titled "Platformer Games" include all three racing games. Bold values show results where models perform significantly better than on all of the games included in their training set. Underlined values show results where the test game was included in the model's training set.

μA				
Test Game	**Endless & Pirates!**	**Endless & Run'N'Gun**	**Pirates! & Run'N'Gun**	**Platformer Games**
TinyCars	66.4±1.6	66.7±1.6	64.8±1.2	66.5±1.6
Solid	68.4±0.9	73.6±0.8	71.4±0.4	73.6±0.8
ApexSpeed	69.5±1.4	71.0±1.5	69.6±1.4	71.0±1.5
Heist!	**77.5±0.8**	78.2±0.7	74.1±0.9	77.5±0.7
TopDown	**79.8±1.1**	73.0±1.1	81.4±1.0	**83.1±1.1**
Shootout	**81.5±1.0**	**82.7±1.1**	**85.0±0.8**	**83.6±1.0**
Endless	69.7±2.0	69.8±2.0	70.0±2.0	69.9±2.0
Pirates!	69.3±1.8	69.5±1.7	69.5±1.7	69.4±1.8
Run'N'Gun	74.6±1.4	79.5±1.9	80.1±1.9	80.0±1.9

∇A				
Test Game	**Endless & Pirates!**	**Endless & Run'N'Gun**	**Pirates! & Run'N'Gun**	**Platformer Games**
TinyCars	54.6±0.6	55.7±0.5	56.4±0.3	54.6±0.5
Solid	55.2±0.5	56.8±0.4	56.6±0.6	56.6±0.6
ApexSpeed	60.5±0.4	63.4±0.5	59.1±0.7	62.8±0.5
Heist!	67.2±0.4	64.7±0.4	**67.0±0.5**	67.2±0.4
TopDown	58.4±0.4	60.2±0.6	62.4±0.7	60.7±0.6
Shootout	50.6±0.5	51.1±0.5	50.9±0.4	51.0±0.4
Endless	70.3±0.9	69.8±0.9	**62.3±0.6**	68.7±0.8
Pirates!	58.9±0.5	56.5±0.4	58.7±0.5	59.1±0.5
Run'N'Gun	53.5±0.5	60.9±0.4	60.6±0.4	60.0±0.5

and Run'N'Gun) by +2%. While in game-based models, shooter games were able to predict Run'N'Gun with very high accuracy, this was not true vice versa. Surprisingly, including at least one other game in the training set significantly improves the performance of models featuring Run'N'Gun. From a 70% and 77% average accuracy on Heist! and TopDown, respectively, the performance jumps to 77% and 79% on average. Interestingly, adding Endless to Run'N'Gun increases the performance on Heist!, while adding Pirates! increases the performance on TopDown. A possible explanation is the difference in gameplay dynamic between these games. Both Heist! and Endless focus more on proximal goals as their levels unfold without much look-ahead. In Pirates! and TopDown, on the other hand, the level is more structured towards a linear progression, and due to the camera's perspective, there is more space visible to the player.

Platformer genre ∇A models have an average accuracy of 58%, 59%, and 62% for the racing, shooter, and platformer games, respectively. The overall individual model performance is 60% with models trained on sets featuring Pirates! doing slightly worse (−1% on

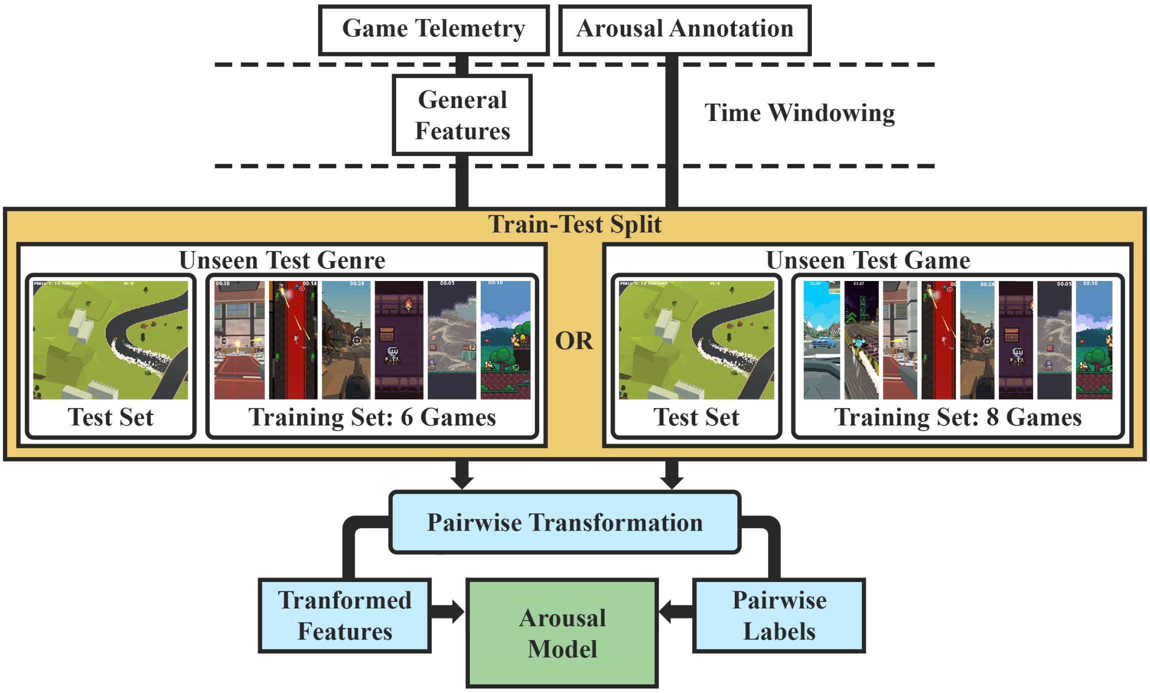
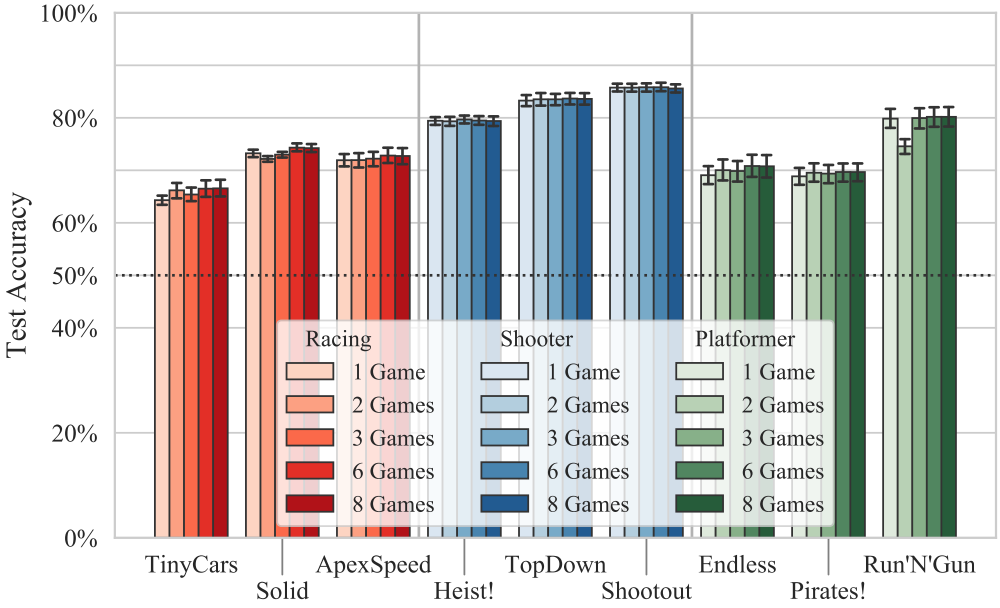


Figure 7.1: General modelling pipeline for modelling arousal. Models for unseen genres are trained on data from six games from two genres and tested on an unseen game of the unseen genre. Models for unseen games are trained on all three games within the dataset except one and tested on the unseen game. Preference learning is applied by using a Pairwise Transformation, in which the ranking problem is reformulated as binary classification of pairwise labels (see Section 2.3 for more details).

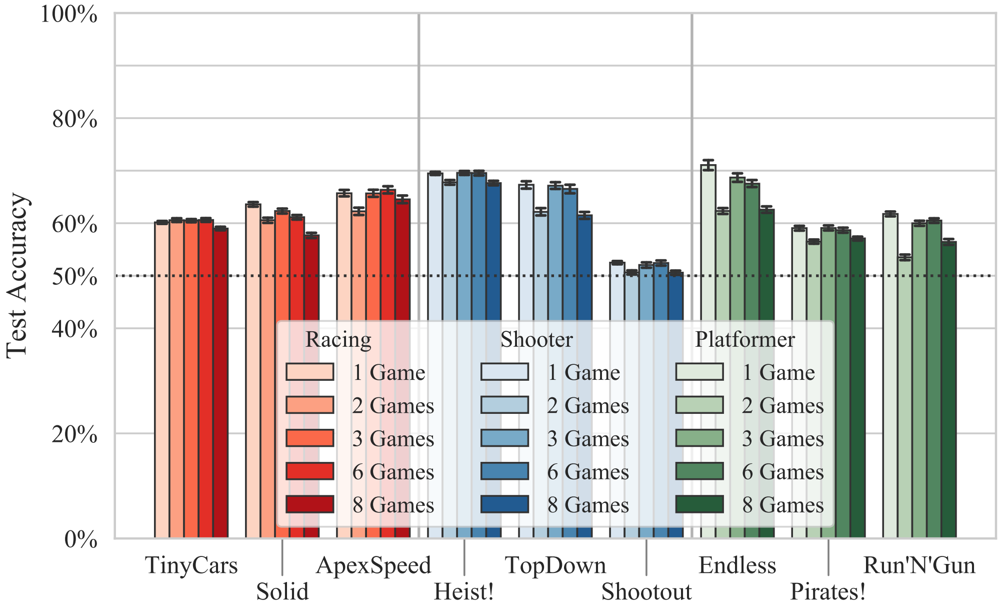
average). Similarly to game-based models, platformer genre ∇A models predict Heist! with higher accuracy than some of the games they were trained on. Interestingly, ∇A models trained on sets containing Pirates! are predicting Heist! with high accuracy, even though μA models trained on similar sets were less successful in predicting Heist! than μA models trained only Endless and Run'N'Gun.

7.3 General Model Performance

Table 7.7 shows the results of general modelling. Within general μA models trained on 6 games have a general performance of 71%, 82%, and 74% when tested on games of the racing, shooter, and platformer genres, respectively. While models trained on 8 games have a marginally higher accuracy on shooter games (83% on average), the two different types of general models have no significant difference in performance. Compared to the genre and game models, there is a significant improvement in the case of the Solid (average +2% from 2 game genre-based models to general ones) and Run'N'Gun (average +6% from 2 game genre-based models to general ones) tests. Although none of the μA models improve significantly on game-based models, the performance of general models is comparable and, in some cases, marginally higher. These results show the robustness of general modelling as games from different genres can reliably model the change in player arousal in most cases. Unsurprisingly, the high performance in these instances is very close to the best matching



(a) Models trained with μ_A arousal.



(b) Models trained with ∇A arousal.

Figure 7.2: Overview of the performance of general models for each game in the dataset. Models are trained on either 6 or 8 games. Models trained on 6 games contain games from two genres and tested on the games of the unseen genre; models trained on 8 games are trained on all but one game and tested on the unseen game. The hue shows the number of training games. Models trained on *general* features from Chapters 5 and 6 are included for comparison (1 Game, 2 Game, and 3 Game models). The dotted line shows the natural baseline and the error bars indicate a 95% confidence interval.

Table 7.7: Testing accuracies (%) of general models trained on 8 and 6 games compared to genre and game models from Chapters 6 and 5. General models trained on 8 games are tested on the unseen game and models trained on 6 games are tested on games of an unseen genre. All models were trained on *general* features. Bold values show the best models for the given game and target output.

μA					
Test Game	**General**		**Genre**		**Game**
	8 games	**6 games**	**3 games**	**2 games**	**1 game**
TinyCars	**66.6±1.7**	66.5±1.7	65.4±1.3	66.2±1.5	64.3±0.9
Solid	74.2±0.7	**74.3±0.8**	73.0±0.5	72.2±0.5	73.2±0.7
ApexSpeed	72.7±1.5	**72.8±1.5**	72.2±1.3	71.9±1.4	71.9±1.3
Heist!	79.4±0.9	79.4±0.9	**79.7±0.8**	79.3±0.9	79.4±0.7
TopDown	**83.6±1.1**	83.2±1.0	83.5±1.1	83.5±1.1	83.3±1.1
Shootout	85.6±0.8	84.1±0.9	85.8±0.8	85.8±0.8	**85.8±0.8**
Endless	**70.8±2.1**	70.8±2.1	69.9±2.1	70.0±2.0	69.1±1.8
Pirates!	69.7±1.7	**69.7±1.8**	69.4±1.7	69.5±1.7	68.9±1.7
Run'N'Gun	**80.2±1.9**	80.2±1.9	80.0±1.9	74.6±1.4	79.8±1.9

∇A					
Test Game	**General**		**Genre**		**Game**
	8 games	**6 games**	**3 games**	**2 games**	**1 game**
TinyCars	59.0±0.3	55.9±0.3	60.5±0.3	**60.6±0.4**	60.2±0.3
Solid	57.7±0.5	56.1±0.7	62.3±0.5	60.6±0.5	**63.6±0.4**
ApexSpeed	64.5±0.7	63.0±0.5	65.7±0.7	62.3±0.7	**65.7±0.6**
Heist!	67.6±0.4	64.1±0.4	**69.6±0.4**	67.8±0.4	69.5±0.3
TopDown	61.5±0.6	60.4±0.6	67.1±0.7	62.2±0.7	**67.3±0.7**
Shootout	50.6±0.4	51.9±0.5	52.1±0.5	50.6±0.4	**52.5±0.3**
Endless	62.6±0.6	63.7±0.6	68.7±0.8	62.3±0.6	**71.1±0.9**
Pirates!	57.1±0.4	57.0±0.4	59.1±0.5	56.5±0.4	**59.1±0.5**
Run'N'Gun	56.4±0.6	58.0±0.5	60.0±0.5	53.5±0.5	**61.8±0.5**

game or game combinations from Sections 7.1 and 7.2. However, results here show that adding more games to the training set does not confuse the models as they remain robust predictors of arousal.

While adding more games to the training set of μA models leads to a steady increase in performance in general, the same is not true for ∇A models. General ∇A models trained on 6 games have a general performance of 60%, 60%, and 59% and models trained on 8 games have a general performance of 58%, 59%, and 60% when tested on games of the racing, shooter, and platformer genres. The difference between these models is significant in 6 out of 9 cases (except TopDown, Endless, and Pirates!). General ∇A models trained on 6 games perform significantly worse than both type of genre-based models (-3% on average), with the exception of ApexSpeed and Pirates!—in case of genre models trained on 2 games—and Shootout—in case of genre models trained on 3 games. The difference between general ∇A models trained on 8 games and genre-based ∇A models is less prominent. When comparing between these general models and genre models trained on 2 games, general models show a significant difference in accuracy only when tested on TinyCars (-2%), Solid(-3%),

Table 7.8: Feature importance of genre models trained on 6 games (two genres) and 8 games (3 genres), as derived from the Random Forests, averaged across games of the same genre. Features present in top five features of all models with the same output are shown in bold.

Genre	μA		∇A	
	Feature	**Score**	**Feature**	**Score**
Racing & Shooter	**Time Passed**	0.378	**Bot Diversity**	0.137
	Player Score	0.261	**Bot Count**	0.102
	Bot Diversity	0.051	Player Score	0.088
	Player Activity	0.050	Time Passed	0.086
	Event Diversity	0.044	Event Diversity	0.085
Racing & Platformer	**Time Passed**	0.322	**Bot Count**	0.124
	Player Score	0.269	**Bot Diversity**	0.106
	Player Movement	0.055	Bot Movement	0.104
	Bot Movement	0.049	Player Movement	0.090
	Player Activity	0.046	Object Diversity	0.084
Shooter & Platformer	**Time Passed**	0.382	**Bot Diversity**	0.159
	Player Score	0.272	Bot Movement	0.148
	Bot Diversity	0.064	**Bot Count**	0.118
	Event Diversity	0.047	Event Diversity	0.093
	Event Intensity	0.041	Event Intensity	0.066
Racing, Shooter & Platformer	**Time Passed**	0.377	**Bot Diversity**	0.140
	Player Score	0.286	**Bot Count**	0.124
	Bot Diversity	0.046	Bot Movement	0.110
	Event Diversity	0.041	Event Diversity	0.082
	Event Intensity	0.039	Player Movement	0.069

ApexSpeed(+2%) and Run'N'Gun(+3%). When comparing between general ∇A models rained on 8 games and genre models trained on 3 games, general models show an average -3% decrease in accuracy. This effect is significant in all tests except on ApexSpeed. Finally, when comparing general ∇A models to game-based ones, general models show an average -5% drop in accuracy. This decrease is significant in almost all cases (except models trained on 8 games and tested on ApexSpeed and models trained on 6 games and tested on Shootout). Results here show that predicting the change in a more complex temporal dynamics of arousal through ∇A is considerably harder than predicting the overall change. The results shown in Table 7.7 correspond to what could be expected from transferring models to predict new elicitors. However, even if adding more games to the training sets of these models do cause some confusion, the trade-off between the general application and decrease in accuracy is relatively low.

7.4 Impact of Features on Model Performance

Similarly to Chapters 5 and 6, this section also examines the feature importance of the models introduced in this chapter. Table 7.8 shows the top five features of general models trained on 6 games—that is, games from two genres—and models trained on 8 games—that is, games from all three genres available in the AGAIN dataset. Unlike in previous chapters, the features presented here are all general features. For a detailed breakdown of all feature importance across general models, see the tables in Appendix A.3.4 for models trained on 6 games and Appendix A.3.5 for models trained on 8 games.

In μA models, a familiar pattern emerges. The absolute top features are Time Passed and Player Score by a good margin over the other features. Similarly to what has been observed in Chapter 6, adding more games to the training set increases the weight of these features in the RF models. While models trained on 1 game had an average of 0.11, models trained on 2 games had an average of 0.15, and models trained on 3 games had an average of 0.17 MDI score for these features, general μA models trained on 6 games have an average of 0.314 and models trained on 8 games have an average of 0.331 score for these features. This observation shows that in arcade-like casual games—such as the ones included in AGAIN—the strong relationship between time and arousal creates a robust predictor regardless of other hand-crafted features. The same shift is not true for ∇A models. However, for the first time, ∇A models also show consistent features between different models with different inputs. Bot Diversity and Bot Count are the most consistently prominent features in this case. While models presented in previous chapters were more reliant on the player's status (e.g. Player Damaged) and action (e.g. Player Speed X), general models encompassing games from multiple genres evidently shift the focus to bot presence. As ∇A measures the change in the temporal dynamics of gameplay, it makes sense that appearing and disappearing enemies affect it.

7.5 Sensitivity to Memory

This section revisits general models presented in this chapter and observes how dynamic memory windowing affects model performance. In Chapters 5 and 6 game and genre models were generally improved when memory was extended to the whole session history with a few exceptions. As evidenced by Table 7.9, this pattern continues with general models as well. General μA models trained on both 6 and 8 games perform significantly better with dynamic memory windowing. The average gain is +11%, +17%, and +15% for the racing, shooter, and platformer games across all models. Just as with game and genre models, the highest gain is with Shootout and Run'N'Gun (+32% and +24% on average, respectively). The increase in accuracy here, however, is considerably higher than in the case of game and genre models. This observation, coupled with the continuous shift in weight towards Time Passed and Player Score features in general models, suggest that the player experience in Shootout and Run'N'Gun are even more sensitive to time than previously hypothesised (see Section 5.4).

While ∇A models also show an overall improvement with dynamic memory windowing, the effect is much smaller and less consistent in a number of ways. First, any performance difference shown on TopDown and Pirates! is not significant. Second, in the case of Shootout and Run'N'Gun, using dynamic memory windows is detrimental to the performance of general models (on average -1% in case of models trained on 6 games and -3% in case of models trained on 8 games). In other cases, the overall average gain when using dynamic memory windowing is +5% across all models. The best improvement is achieved on ApexSpeed (on average +8% when models are trained on 6 games and +10% when models are trained on 8 games). This is a surprising result as both in the case of game-based models and genre-based models trained on 3 games, models trained with consecutive windowing performed better on ApexSpeed. However, in those cases, the training set of the models also included this game. This observation suggests that general models not

Table 7.9: Testing accuracies (%) of general models trained on 6 and 8 games and tested on an unseen game with consecutive time-windows (consecutive windows) compared to best models trained with a dynamic memory window (full history). Bold values show the best models for the given game, training set, and target output.

μA				
Test Game	**8 games**		**6 games**	
	Consecutive	**Full History**	**Consecutive**	**Full History**
TinyCars	59.0±0.8	**66.6±1.7**	58.5±0.9	**66.5±1.7**
Solid	62.8±0.9	**74.2±0.7**	60.9±0.7	**74.3±0.8**
ApexSpeed	62.7±0.9	**72.7±1.5**	59.0±1.1	**72.8±1.5**
Heist!	70.7±0.7	**79.4±0.9**	68.9±0.7	**79.4±0.9**
TopDown	76.9±0.5	**83.6±1.1**	74.7±0.6	**83.2±1.0**
Shootout	52.6±1.0	**85.6±0.8**	52.2±0.9	**84.1±0.9**
Endless	61.8±1.0	**70.8±2.1**	63.3±1.0	**70.8±2.1**
Pirates!	58.1±0.8	**69.7±1.7**	56.8±0.8	**69.7±1.8**
Run'N'Gun	56.8±1.0	**80.2±1.9**	56.3±1.0	**80.2±1.9**

∇A				
Test Game	**8 games**		**6 games**	
	Consecutive	**Full History**	**Consecutive**	**Full History**
TinyCars	52.9±0.4	**59.0±0.3**	51.5±0.3	**55.9±0.3**
Solid	52.6±0.3	**57.7±0.5**	52.0±0.3	**56.1±0.7**
ApexSpeed	54.9±0.4	**64.5±0.7**	55.4±0.4	**63.0±0.5**
Heist!	62.1±0.4	**67.6±0.4**	61.2±0.4	**64.1±0.4**
TopDown	**62.6±0.6**	61.5±0.6	**61.5±0.5**	60.4±0.6
Shootout	**54.1±0.5**	50.6±0.4	**53.4±0.5**	51.9±0.5
Endless	57.4±0.3	**62.6±0.6**	56.8±0.2	**63.7±0.6**
Pirates!	**57.3±0.4**	57.1±0.4	56.9±0.4	**57.0±0.4**
Run'N'Gun	**58.0±0.4**	56.4±0.6	**58.9±0.4**	58.0±0.5

trained on the target game can benefit greatly from the dynamic memory window during the pairwise transformation in preference learning models. All in all, these experiments have shown the robustness of the asymmetric and dynamic memory windowing method. While the improved results of μA could be attributed to this output's strong correlation with time, results of ∇A models show that a dynamic memory window can be beneficial regardless of this connection.

7.6 Summary

This chapter started with an overview of the generality of game-based and genre-based models. Results showed that gameplay dynamics play a prominent role in what games are compatible regardless of genre. While in some cases stacking more games in the training set led to lower performance compared to models training on single games, overall, adding more games to the training set had been proven to be beneficial. The second half of this chapter presented two approaches to general arousal modelling. In the first case, one genre has been left out of training sets to test the general robustness of models across different genres. In the second case, one game has been left out of training sets to test the upper limit of the performance of general models on unseen games. While the former method shows how general models can be applied to entirely new games, the latter maximises the information fed to the models. Just as before, models were tested with two different outputs. In the case of general μA models trained on games from two genres (6 games) had a general performance of 71%, 82%, and 74% when tested on games of the racing, shooter, and platformer genres respectively. General μA models trained on 8 games could only improve the model performance on shooter games significantly (83% on average). General ∇A models trained on 6 games had an average accuracy of 60%, 60%, and 59% and models trained on 8 games had a average accuracy of 58%, 59%, and 60% when tested on games of the racing, shooter, and platformer genres, respectively. Subsequent analysis of feature importance revealed an even stronger reliance on time-related features than shown in Chapters 5 and 6 in the case of μA models. The same analysis also showed that in the case of ∇A models, Bot Diversity and Bot Count emerges as consistently robust features. Finally, a sensitivity analysis of the windowing method applied to the data showed that general models could largely benefit from using dynamic memory windows to encompass the available session history in preference learning models leveraging a pairwise transformation.

The next chapter concludes this thesis by revisiting the research questions posed in Chapter 1. This chapter also presents the main contributions of the work, limitations of the projects, and future extensibility of the research presented here.

Chapter 8

Discussion and Conclusions

This thesis set out to investigate the feasibility of reliably predicting player affect in unseen games. To this end, studies presented here focused on modelling player *arousal* through game telemetry data in a set of casual games designed for this purpose. This work included tools enabling a robust pipeline for first-person affect annotation in videogames, a large dataset of gameplay telemetry and player arousal, and predictive models across different levels of generality. Because there were no datasets in the field containing multiple games within the same genres, collecting new data was necessary to complete the thesis. The result of this project and a main contribution of the thesis is the Affect Game Annotation Dataset (see Chapter 4), which includes gameplay telemetry and arousal annotation that allowed to address the core research question. Based on the collected data, preference learning models have been constructed using pairwise transformation and Random Forests. Early on in Chapter 1 the investigation was framed by two sets of research questions relating to these aforementioned models and procedures. This chapter reexamines these research questions and attempts to answer them in light of the key findings of this thesis.

The first research question, "Can *arousal* in games be reliably modelled in a general fashion?" is composed of three sub-questions that led the exploration on three different levels of generality: game-based modelling using game-agnostic features; genre-based modelling leveraging genre-specific and general features; and finally general models, which can be applied to previously unseen games of any genre.

- **RQ1.1: Can game-agnostic features perform comparably to specific features?** Chapter 5 set out to answer this question. Results of this chapter showed the robustness of hand-crafted heuristic features. Game-based models trained on general features showed comparable performance to models trained on genre-specific features. This was true for models both predicting a change in the level of arousal and—to a lesser extent—a change in the gradient of the arousal curve.

- **RQ1.2: Can models be transferred between games within a genre?** Chaper 6 aimed to answer this question by investigating genre-based modelling. Models in this chapter were trained on multiple games within a genre and tested on both seen and unseen games. Results showed that models trained on multiple games within a genre could reliably predict games of the same genre regardless of whether the algorithm saw them during training. In some cases, using multiple games in the training set improved the performance beyond that of game-based models. Chapter 7 followed up on this investigation by looking at the transferability of game-based and genre-based models. Results showed that shooter games (except for Shootout) were the most robust predictors among both game- and genre-based models. An explanation for this phenomenon is the relative complexity of these games compared to others in the dataset.

- **RQ1.3: Can models generalise over games from different genres?** Chapter 7 focused primarily on answering this question. Results in this chapter showed great potential for general arousal modelling across different genres by investigating both genre- and game-agnostic approaches. The best results were comparable to the game-based model baselines. Analysis of feature importance in this and previous chapters showed that the models predicting the change in the level of arousal rely primarily on time-related features, while models predicting the change in the gradient of the arousal trace rely more on telemetry features relating to bots.

The second research question, "How to reliably capture and model *arousal* in games?" was also sub-divided into three smaller questions, which focused on the pipeline of modelling, mitigating memory effects, and modelling more complex temporal dynamics beyond a relative change in affect.

- **RQ2.1: How can we capture the first-person impression of arousal during gameplay?** Chapter 3 aimed to answer this question by proposing an affect annotation pipeline for videogame research using stimulated recall techniques and first-person time-continuous annotation. This pipeline builds on the *Platform for Audiovisual General-purpose ANnotation*, which was developed for the collection of affective labels. The proposed procedure consists of blocks of gameplay followed immediately by an annotation task of the same play session. This setup allows for the easy crowd-sourcing of both gameplay and labelling tasks while also minimising interruptions during the process. As the data is collected within the same framework, it can be easily synchronised and pre-processed for subsequent modelling.

- **RQ2.2: Can temporal biases be mitigated during data processing?** This question is answered through the analysis presented in Chapters 5-7. It was shown that through a dynamic time-windowing method introduced in Chapter 2, the accuracy of models could be greatly improved. The proposed methodology builds on a standard pairwise transformation; a pre-processing step applied in particular ranking approaches. As many models rely on time-related features, this processing method can mitigate some of the recency effects of annotation. Through a subsequent analysis of the results it was shown that applying a dynamic memory window is beneficial for models where there is no strong correlation between time and the affective output as well.

- **RQ2.3: To what extent can the temporal dynamics of the annotation be modelled?** To answer this question, models throughout Chapters 5-7 have been constructed to predict both the change in the level of arousal and the change in the gradient of the arousal trace. As the gradient of a signal describes the direction and the rate of the increase or decrease of the trace, it carries more complex information about the temporal dynamics of gameplay than the mean values of time-windows. Results throughout the thesis showed that predicting the change in the gradient of arousal is a significantly more challenging task than predicting just the increase or decrease in arousal. As this ground-truth did not have any temporal bias, these models could not rely on game-agnostic time-related features. Subsequently, the models were not as successful in generalising to games other than what they were trained on.

While answering the primary question of this thesis—"How to reliably predict player affect in any unseen game?"—is still an open question for the future, the first-person annotation, pre-processing, and modelling pipeline presented in this thesis takes a good step towards answering this inquiry. The rest of this chapter is dedicated to a discussion on the primary contributions of this thesis (Section 8.1), the limitations of the presented projects (Section 8.2), and the extensibility of the work presented here through future research (Section 8.3). Finally, Section 8.4 summarises the discussion and concludes the thesis.

8.1 Contributions

This section provides an overview of the different contributions of this thesis to the fields of player modelling and affective computing. While the work presented here focuses on videogame play, the tools produced and lessons learned can easily be adapted in other domains where the goal is time-continuous affect or emotion prediction. The following list details the main contributions of this thesis in more detail:

- **Pipelines for First-Person Affect Annotation in Videogames**: This thesis presented a streamlined pipeline for the first-person affect annotation in videogames. The *Platform for Audiovisual General-purpose ANnotation* (PAGAN) (Melhart et al., 2019b) was designed to enable the crowd-sourcing of the game playing and annotation tasks in this thesis work. PAGAN offers an open-source online annotation tool featuring multiple popular labelling methods for any multimedia content. It was designed to be user-friendly and does not require any programming knowledge to allow for the tool's adoption in fields beyond affective computing. Since its introduction PAGAN has already been used in a diverse array of studies from deep-learning (Makantasis et al., 2021) through musical sentiment-analysis (Gulati et al., 2020) and attention research in linguistics (Ravelli et al., 2020) to annotation processing in affective computing (Booth, 2020). This thesis built on this system to develop a first-person affect annotation pipeline for videogame research, which minimises the interruptions during the process. The procedures presented in Chapter 3 show this robust pipeline from data collection to pre-processing. This pipeline was used to collect the dataset presented in Chapter 4 and is being already adapted in studies like the aforementioned (Makantasis et al., 2021) and time-continuous believability modelling (Pacheco et al., in review).

- **Affect Game Annotation Dataset**: This thesis also introduced the *Affect Game Annotation Dataset* (AGAIN) (Melhart et al., 2021). This database was created in response to the lack of comprehensive and well-structured datasets, including multiple games or interactive elicitors in affective computing. AGAIN contains 9 games from 3 genres, 124 participants, and over $1,100$ annotated gameplay sessions. While the fields of games user research and affective computing often work on similar problems, the ad-hoc nature of game datasets made it hard to bridge some of the gaps between these fields. As most datasets generally focused on one or two games, they were unfeasible for the comprehensive study of general modelling. Beyond this motivation, AGAIN also enables future research into the application of deep-learning in the field of affective computing and player modelling as the database also contains more than 37 hours of gameplay video footage.

- **General Preference Learning Models**: Studies presented in this thesis work provide a comprehensive overview of hand-crafted general features, their robustness and prominence in player models from game-based through genre-based to general applications. Analysis throughout the presented chapters shows some of the opportunities and pitfalls of focusing on casual and arcade-like testbed games. The strong correlation between the time passed in games and the intensity of the gameplay can help the modelling effort of similarly structured games and proves the feasibility of general modelling in this domain. However, it also highlights some of the limitations of the AGAIN dataset and a large portion of general game AI and experimental player research as these studies often employ similar arcade-style and small-scale casual games.

  This thesis work used pairwise Preference Learning (PL) to build general models for the time-continuous prediction of player arousal. While PL has been applied to similar problems before, this thesis work introduced a novel processing method for pairwise PL. The dynamic windowing of past time-windows to represent the session history during the transformation of the dataset has proven to be especially useful for general modelling. The presented methodology is very cost-effective, easy to implement, and can be used with virtually any binary classifier. The final results of this thesis show the feasibility of general modelling and provide a clean pipeline and strong baseline for future studies.

8.2 Limitations

This section details the limitations of the work presented throughout the thesis. One general limitation of the work comes from the wicked nature of design and development work; that is, there is no correct answer to a technical problem, just one that is good enough. During the development of the tools and elicitors included in this thesis, decisions had to be made to prioritise certain elements and cut others. Because the dataset had to be collected online (partially due to the 2020 Coronavirus outbreak), there were limitations on the breadth of the data that could be collected. While an effort has been made to deliberate on each of the design choices, they still encode some unseen biases that contribute to the inherent noise of the collected data and the constructed models.

- **Platform for Audiovisual General-purpose ANnotation**: While the tool has already integrated three different kinds of annotation frameworks, all of these methods are time-continuous and one-dimensional approaches. As the tool was designed with the first-person annotation of affect in mind, there is no categorical labelling or video-scraping feature in the tool, which would allow expert annotators to move forward and backward on the video, adding, removing, and updating both continuous and categorical labels. While there is support for integrating Google Forms[1] at the end of an annotation protocol, currently, there is no support for using survey methods between annotation sessions within the same protocol. Finally, although the online interface provides an out-of-box solution using the YouTube API[2], the open format

[1] `https://www.google.com/forms/about/`

[2] `https://developers.google.com/youtube/`

comes with some caveats. Namely, there is no way for handling confidential data, for example, the participants' likeness or footage under non-disclosure agreements. While it is possible to set up a custom version of the platform on a private server using the source code of the application[3], setting up a version of the system like this does require at least a rudimentary knowledge of web development.

- **Affect Game Annotation Dataset**: Even though the AGAIN dataset is one of the largest open-source videogame databases with affective annotations out there, it also has several limitations. While the games included in the dataset cover a good portion of casual games, major popular genres are missing from the dataset. While popular on the console, PC, and mobile markets, role-playing, strategy, and puzzle games are all absent from AGAIN. As it has been mentioned before in Chapter 4, the omission of these genres from the dataset was partly a technical decision. However, as it has been shown in Chapters 5-7, the focus on casual games come with some caveats. While these games can be considered approximations of some mobile games, which are meant to be played in short bursts, games in the console and PC market have a very different dynamic. In these games, there are more robust gameplay mechanics, often with a more pronounced learning curve. Unfortunately, as gameplay sessions in AGAIN are constrained to a two-minute length, they cannot capture these dynamics. Another major limitation of AGAIN is the focus on only one affective dimension. While other datasets generally include at least two affective dimensions (see Chapter 2), AGAIN only includes arousal. Even though arousal as an approximation of perceived gameplay intensity makes sense as the primary focus of such a dataset, it also makes AGAIN less diverse than its counterparts. While the dataset includes telemetry and gameplay video, peripheral and physiological signals are missing. This was an unfortunate drawback of the necessary crowd-sourced collection of the data. Finally, as AGAIN focuses on interactive elicitors, each session is unique. Because AGAIN uses first-person annotation, this also means that there is only one annotation trace per elicitor. While the first-person annotation does provide a certain level of reliability and validity to the dataset, it also prevents the usage of a wide array of techniques commonly used in affective computing to clean and process labels.

- **Affect Modelling**: The presented arousal models, while often reach a surprising level of accuracy, are somewhat simplistic. Firstly, the input space of the models is based on hand-crafted telemetry features. While this focus allows for a transparent feasibility study as each feature is human-readable and meaningful, collecting telemetry data in the wild is often costly or unfeasible. Secondly, while Random Forests (RF) are shown to be robust predictors, looking at the performance of different models, it is clear that results likely cannot be improved further through fine-tuning RF parameters. Finally, the presented models do not use the recorded video footage included in the AGAIN dataset either.

8.3 Extensiblity

This section focuses on the possibilities to extend the presented tools and the research as a whole to different domains and applications.

[3] `https://github.com/davidmelhart/PAGAN`

8.3.1 Extending PAGAN

Future work on PAGAN includes improving the system's flexibility with options for video-scraping, categorical labels, forced-choice questionnaires, and other survey steps. These extensions would allow a larger number of options which would open up more complex research protocols. As it has been demonstrated in Chapter 2, while there is a need for flexible tools for both dimensional and categorical labelling, many of these tools are out of date. This gap could be filled more sustainably with PAGAN as the tool is based on web technology, making it easier to access, modify, and share than many other program packages. While some annotation tools move towards more robust machine learning modules (Heimerl et al., 2019), future extensions would push the features of PAGAN as a lightweight but comprehensive annotation framework for traditional labelling and quick prototyping. As web cameras are readily available in almost all devices, another extension of PAGAN could focus on gathering additional information like gaze direction and facial action units. This extension would allow for the collection of additional peripheral information not just in a lab setting but also in a crowd-sourced manner. Finally, issues with confidential information can be solved by creating an offline version of PAGAN, which would allow an out-of-box deployment in a local lab environment. This would allow for even more widespread adoption of PAGAN.

Instead of extending PAGAN itself with machine learning capabilities, the output of the platform could be connected to the Python Preference Learning Toolbox (pyPLT) (see Chapter 2). While PAGAN and pyPLT are written in different languages, synchronising the data formatting and pre-processing and implementing a shared web API could go a long way towards integrating these systems. In the future, data captured by PAGAN could be reached directly in pyPLT, and with a shared pipeline in place, acquiring a trained model would be as easy as pressing a button, with no additional processing needed.

8.3.2 Extending AGAIN

Some of the current limitations of AGAIN can be addressed with additional data collection, while others need new elicitors and collection protocol to be implemented. Such as the issue of new game genres and games which mimic a console or PC-game experience better with longer sessions and more varied intensity. Fortunately, both the lack of other affective dimensions in the dataset and the uniqueness of the elicitors can be solved to some degree by simply collecting more data using the existing videos. Third-person annotators can label sessions with arousal, valence, and additional affective dimensions to create a more robust dataset.

With the extension of PAGAN, new opportunities arise to update AGAIN as well. A new AGAIN dataset would include gaze information and facial action units as well, and an in-lab version of PAGAN could facilitate the collection of psychophysiological signals. Such a dataset would be a more rounded version of AGAIN with more possibilities for affective computing research into visual attention, emotion elicitation and recognition, and affective physiology.

8.3.3 General Deep-Learning and Temporal Modelling

Many possibilities of extending the presented research to other areas would rely on finding new general features. While physiological signals are commonly used in affective computing as elicitor-agnostic features, they are often unfeasible to collect. Therefore, finding generalised end-to-end solutions should also focus on more readily available raw data. An excellent solution to address the limitations of the current affective models in this light would be applying deep-learning via neural networks using the video data included in the AGAIN dataset. Deep learning applications could learn from the pixel information to build end-to-end models of the presented games. Deep-learning models could still use privileged information encoded by telemetry features; however, they would make the models more feasible for industrial application. Additionally, general features can also be found using unsupervised transfer learning to find a better mapping between different games. While the models presented in this thesis work all use similar features (even genre-specific features across genres), raw visual input is far more varied. Even though general pipelines have been investigated before (Makantasis et al., 2021), successful general end-to-end emotion modelling is still an open problem.

Even though the presented models are not geared towards computer vision, a possible contribution of the research presented here lies in the approach to pairwise preference learning. Studies in this thesis showed that aggregating past datapoints to an abstract anchor could significantly improve model performance. While preference learning long- and short-term memory (LSTM) neural networks might provide a more complex representation of how the users' long-term preferences develop (Wu et al., 2020), the presented research shows a simple, cost-effective, and robust approach that can be easily adapted in more naive algorithms as well. Future research should focus on testing the proposed temporal aggregation method against and in combination with LSTM systems.

8.3.4 Procedural Content Generation and Adaptive Design

Within game research, the study of general affect modelling can be extended to more immediate game-related emotional outcomes, such as engagement or tension. Such predictions are easier to use for designers as input for adaptive and generative systems. General models could also help jump-start more extensive research into procedural systems using an *affective loop* (Yannakakis and Togelius, 2018). These systems could produce content and adapt to personal experiences based on the predicted emotional outcome of players or could be used to author emotional content to the designer's specification. Unfortunately, there is a high cost associated with such solutions. On top of creating a traditional game, emotional models have to be developed and connected to the larger design of the product. This investment is often too high for commercial application of these models, but it also slows down academic research. With reliable general models, the cost of developing these affective systems could decrease. A largely plug-and-play model, which can provide reliable emotional predictions of unseen games, could lead to a more widespread adaptation of affective systems in videogame development and research.

8.3.5 Domains of Affective Computing

As the presented research focuses on time-continuous behavioural telemetry, the same approach can be adapted to other domains of human-computer interaction with relative ease. Even though traditional affective computing applications often focus on passive multimedia elicitors (see Chapter 2), a time-continuous model of affect can be built using physiological and peripheral signals as input for the models. While studying the emotional effects of multimedia clips presents a more homogeneous domain than games, models based on physiological and peripheral signals could virtually bridge different media types.

End-to-end pixel-to-emotion models trained on videogames can be used to predict the emotional outcome of passive audiovisual experiences, while models trained on psychophysiological signals could predict the emotional content of a text as well. These general models can also provide a baseline prediction for other affective computing applications in education (Wu et al., 2016; Yadegaridehkordi et al., 2019) and healthcare (Yannakakis, 2018). In the former case, general models of arousal can be used to monitor learners' engagement and attention cost-effectively. This is especially important in today's world of distance education, where traditional methods of monitoring and interacting with the learner is not available. Similarly to how the affective models are built in this thesis, general models of cognitive processes can also be developed. Generalised models of motivation and attention could enhance and help personalise educational content over a broad spectrum of domains and different types of learning material. In research into healthcare and rehabilitation, a general baseline model of arousal can be used to detect outliers signifying potential health issues. As generalised models can adapt to different elicitors in the wild, they could significantly enhance stress detection as well. In areas where games have already been proven to be valuable tools for therapy (e.g. post-traumatic stress disorder (Holmgård et al., 2013)), general models can be trained in game-like environments and applied in the wild. Similarly, general models can be put to use helping people with autism or similar developmental disorders (El Kaliouby et al., 2006). These models can adapt to unexpected situations and either help interpret dissimilar scenarios or communicate the user's state to others (Han et al., 2018). Finally, the domain of social robots can also benefit from a generalised understanding of affective human behaviour (Van de Perre et al., 2018). These general models could be adapted in different robots, alleviating the need for specialised implementations. Through a general understanding, robots can assess the emotional content of a situation under different circumstances and communicate with humans about different emotional elicitors. These applications would work similarly to the concept of *affective loop* in game research, where an independently trained emotion processing module could be fit into a complex system, which makes use of the emotion prediction by adapting its behaviour to either facilitate or shift the predicted emotional response.

8.4 Summary

This thesis asked the question: *to which degree can we predict player affect in unseen games?* and presented a comprehensive data collection, annotation, and modelling pipeline to answer this question. During this thesis, work tools have been developed to capture the first-person impression of arousal from players. These tools helped the collection of one of the most extensive affective datasets in games to this day. This dataset allowed for a methodical approach to player modelling, where the question of generality was observed on three different levels. The presented studies showed that player arousal could be modelled reliably in a general fashion in casual games. This chapter concluded this investigation by detailing the main contributions of this thesis (Section 8.1), which involved the development of first-person annotation pipelines for interactive elicitors; the design of PAGAN, an online annotation framework for the crowd-sourced collection of time-continuous labels; AGAIN, a sizeable comprehensive dataset of videogame footage, telemetry, and affective labels; and an analysis of the feasibility of general arousal modelling in videogames. Section 8.2 presented some of the limitations of this research, most prominently, the focus on one affective dimension and reliance on telemetry and hand-crafted features. Finally, Section 8.3 aimed to address these limitations and show new avenues for the extensibility of the research beyond this thesis.

Appendix A

Appendices

A.1 Genre-Specific Game Telemetry Features

This section contains the full list of the genre-specific features in AGAIN. Tables A.1, A.2, and A.3 show the racing, shooter, and platformer genre-specific features, respectively. Features marked with 'x' on tables in this section have a non-zero variance in the given game.

Table A.1: Racing genre-specific features.

Feature	TinyCars	Solid	ApexSpeed
Player Standing	x	x	x
Player Speed	x	x	x
Player Speed Boost	-	-	x
Player Grounded	x	x	x
Player Mid Air	x	x	x
Player Looping	-	x	-
Player Collision	x	x	x
Player Off Road	x	x	-
Player Gas Pedal	x	x	x
Player Steering	x	x	x
Player Lap	x	x	x
Player Distance To Way Point	x	x	x
Player Δ Rotation	x	x	x
Player Respawn	x	x	x
Bot Standing	x	x	x
Bot Player Score	x	x	x
Bot Speed	x	x	x
Bot Speed Boost	-	-	x
Bot Grounded	x	x	x
Bot Looping	-	x	-
Bot Off Road	x	x	-
Bot Collision	x	x	x
Bot Gas Pedal	x	x	x
Bot Steering	x	x	x
Bot Lap	x	x	x
Bot Distance To Way Point	x	x	x

Bot Δ Rotation	x	x	x
Bot Player Distance	x	x	x
Bot Respawn	x	x	x
Jump Count	x	-	x
Speed Boost Count	-	-	x
Obstacle Count	-	-	x
Loop Count	-	x	-

Table A.2: Shooter genre-specific features.

Feature	**Heist!**	**TopDown**	**Shootout**
Player Kill Count	x	x	x
Player Speed X	x	x	-
Player Speed Y	x	x	-
Player Speed Z	x	x	-
Player Δ Rotation	x	x	x
Player Health	x	x	-
Player Healing	x	-	x
Player Damaged	x	x	x
Player Shooting	x	x	x
Player Reloading	x	-	x
Player Projectile Count	x	x	x
Player Projectile Distance	x	x	x
Reticle Δ Distance	x	x	x
Player Crouching	x	-	-
Player Sprinting	x	-	-
Player Aim (Enemy)	x	x	x
Player Aim (Destructible)	-	x	-
Pickup (Health)	-	x	-
Bot Speed X	x	x	-
Bot Speed Y	x	x	-
Bot Speed Z	x	x	-
Bot Δ Rotation	x	x	-
Bot Health	x	x	x
Bot Damaged	x	x	x
Bot Shooting	x	x	x
Bot Reloading	-	-	x
Bot Projectile Count	x	x	-
Bot Projectile Player Distance	x	x	-
Bot Aim (Player)	x	x	x
Pickup Count	-	x	-
Pickup Distance	-	x	-
Destructible Count	-	x	-
Objects Destroyed	-	x	-
Player Death	x	x	-
Player Shoot On Reload	x	-	x

Table A.3: Platformer genre-specific features.

Feature	**Endless**	**Pirates!**	**Run'N'Gun**
Player Collision	x	x	x
Player Collision Above	x	x	x
Player Collision Below	x	x	x
Player Collision Left	-	x	x
Player Collision Right	-	x	x
Player Falling	x	x	x
Player Grounded	x	x	x
Player Jumping	-	x	x
Player Speed X	x	x	x
Player Speed Y	x	x	x
Player Health	x	x	x
Player Damaged	x	x	x
Player Shooting	x	-	x
Player Projectile Count	-	-	x
Player Projectile Distance	-	-	x
Pickup (Health)	-	-	x
Pickup (Point)	x	x	-
Pickup (Power)	-	x	-
Pickup (Boost)	x	-	-
Pickup (Slow)	x	-	-
Player on Power	-	x	-
Player Kill Count	x	x	x
Bot Has Collision	-	x	x
Bot Collision Above	-	-	x
Bot Collision Below	x	x	x
Bot Collision Left	x	x	x
Bot Collision Right	-	x	x
Bot Falling	-	x	x
Bot Grounded	x	x	x
Bot Jumping	-	-	x
Bot Speed X	x	x	x
Bot Speed Y	-	x	x
Bot Health	x	x	x
Bot Damaged	-	-	x
Bot Shooting	-	-	x
Bot Projectile Count	-	-	x
Bot Charging	-	-	x
Bot Player Distance	x	x	x
Bot Projectile Player Distance	-	-	x
Pickup Count	x	x	x
Pickup Distance	x	x	x
Player Death	x	x	x

A.2 Correlation Analysis

This section details the results of the correlation analysis on all features within the dataset for each game and genre. The tables are sorted by the best genre-based values. Tables A.4, A.5 and A.6 show the results between the μA and ∇A of arousal and game telemetry features for racing, shooter, and platformer games respectively. Rows marked with 'G' indicate general features, and rows marked with 'S' indicate genre-specific features throughout the tables in this section.

Table A.4: Kendall's τ correlation between racing features and arousal.

μA						∇A					
	Feature	**Racing Genre**	**TinyCars**	**Solid**	**ApexSpeed**		**Feature**	**Racing Genre**	**TinyCars**	**Solid**	**ApexSpeed**
G	Player Score	0.166	0.127	0.215	0.157	S	Player Lap	-0.069	-0.052	-0.082	-0.061
G	Time Passed	0.163	0.117	0.212	0.163	G	Time Passed	-0.068	-0.054	-0.080	-0.071
S	Player Lap	0.125	0.072	0.175	0.135	G	Bot Count	0.066	0.075	0.050	0.043
G	Bot Diversity	-0.097	-0.040	-0.086	-0.186	G	Bot Diversity	0.065	0.069	0.057	0.043
S	Player Speed	0.093	0.077	0.099	0.114	S	Bot Grounded	0.065	0.065	0.058	0.041
G	Bot Count	-0.092	-0.039	-0.089	-0.186	G	Player Score	-0.061	-0.049	-0.072	-0.067
S	Bot Grounded	-0.091	-0.030	-0.091	-0.195	S	Bot Speed	0.057	0.052	0.037	0.041
S	Bot Dist. To WP[1]	-0.089	-0.010	-0.077	-0.187	S	Bot Gas Pedal	0.057	0.052	0.048	0.041
S	Player Gas Pedal	0.088	0.085	0.089	0.087	G	Bot Movement	0.057	0.049	0.037	0.042
S	Player Looping	0.078	0.000	0.148	0.000	S	Bot Dist. To WP	0.054	0.053	0.039	0.044
S	Bot Lap	0.077	0.072	0.144	0.004	G	Player Movement	0.053	-0.005	0.025	0.086
G	Event Intensity	0.073	0.014	0.084	0.125	S	Bot Δ Rot.	0.053	0.052	0.033	0.040
G	Event Diversity	0.069	0.014	0.078	0.121	S	Player Collision	0.052	0.045	0.080	0.042
G	Player Movement	0.065	0.067	0.100	0.063	S	Bot Collision	0.052	0.054	0.055	0.032
S	Player Mid Air	0.065	0.030	0.094	0.123	S	Player Standing	0.050	0.031	0.045	0.036
S	Player Grounded	-0.064	-0.030	-0.080	-0.123	S	Bot Standing	0.048	0.044	0.038	0.063

[1] Way Point

S	Bot Gas Pedal	-0.062	0.006	-0.046	-0.167	S	Player Grounded	0.041	0.008	0.023	0.037
S	Bot Player Dist.	-0.061	0.007	-0.025	-0.178	S	Player Looping	0.041	0.000	0.071	0.000
S	Player Δ Rot.	0.060	0.062	0.097	0.032	S	Player Mid Air	-0.040	-0.008	-0.007	-0.037
S	Bot Δ Rot.	-0.058	-0.002	-0.039	-0.150	S	Player Δ Rot.	0.036	0.022	0.052	0.012
G	Bot Movement	-0.054	0.007	-0.036	-0.165	S	Bot Player Dist.	0.035	0.019	-0.002	0.040
S	Bot Speed	-0.054	0.011	-0.037	-0.163	S	Player Speed Boost	0.035	0.000	0.000	0.126
G	Player Activity	0.049	0.087	0.097	0.033	S	Bot Steering	0.027	0.017	0.004	0.032
S	Player Speed Boost	0.045	0.000	0.000	0.070	S	Obstacle Count	-0.026	0.000	0.000	0.082
G	Input Intensity	0.039	0.030	0.049	0.042	S	Player Steering	-0.025	0.041	-0.046	-0.060
S	Bot Looping	0.037	0.000	0.071	0.000	S	Bot Lap	-0.023	-0.007	-0.053	-0.026
G	Input Diversity	0.036	0.042	0.041	0.032	S	Bot Score	0.022	0.023	-0.009	0.006
S	Bot Steering	-0.034	-0.037	-0.015	-0.052	S	Player Speed	0.020	-0.009	0.022	0.019
S	Bot Collision	-0.034	-0.012	-0.040	-0.050	S	Player Gas Pedal	-0.019	-0.005	0.006	-0.006
S	Bot Score	0.033	0.068	0.090	-0.048	G	Player Activity	0.019	-0.013	-0.007	-0.019
S	Jump Count	0.033	0.011	0.000	0.029	S	Player Respawn	-0.019	-0.028	-0.004	-0.027
S	Player Collision	0.028	0.002	0.057	0.025	S	Loop Count	0.012	0.000	0.015	0.000
S	Player Standing	-0.025	-0.018	-0.029	-0.007	S	Player Dist. To WP	-0.012	-0.043	-0.037	0.049
S	Bot Speed Boost	-0.018	0.000	0.000	-0.038	G	Object Intensity	0.011	-0.003	0.015	0.074
S	Speed Boost Count	0.015	0.000	0.000	0.004	S	Jump Count	0.011	-0.003	0.000	0.078
G	Object Intensity	0.012	0.011	0.020	0.016	G	Event Diversity	0.009	0.011	0.016	0.056
S	Bot Off Road	0.011	-0.001	0.029	0.000	G	Input Diversity	0.008	-0.007	0.016	-0.020
S	Player Off Road	0.008	0.006	0.035	0.000	S	Speed Boost Count	0.008	0.000	0.000	0.069
S	Loop Count	-0.006	0.000	0.020	0.000	S	Bot Speed Boost	0.008	0.000	0.000	0.031
S	Bot Standing	0.006	0.050	-0.004	-0.030	S	Bot Looping	0.006	0.000	0.009	0.000
S	Bot Respawn	-0.006	-0.003	0.018	-0.015	G	Object Diversity	0.004	-0.003	0.018	0.058
S	Player Dist. To WP	-0.005	0.017	-0.041	-0.005	S	Bot Off Road	-0.003	-0.004	-0.012	0.000
S	Obstacle Count	0.005	0.000	0.000	-0.023	G	Event Intensity	-0.002	0.010	0.010	0.030
S	Player Respawn	-0.004	-0.001	0.009	-0.018	S	Player Off Road	0.002	-0.003	-0.046	0.000
G	Object Diversity	0.004	0.012	0.017	-0.031	S	Bot Respawn	-0.001	-0.058	0.020	0.040
S	Player Steering	0.002	-0.009	0.015	0.000	G	Input Intensity	-0.001	-0.013	0.014	-0.064

Table A.5: Kendall's τ correlation between shooter features and arousal.

μA						∇A					
	Feature	**Shooter Genre**	**Heist!**	**TopDown**	**Shootout**		**Feature**	**Shooter Genre**	**Heist!**	**TopDown**	**Shootout**
G	Time Passed	0.363	0.280	0.372	0.445	S	Bot Proj. Count	0.129	0.144	0.166	0.000
G	Player Score	0.338	0.298	0.359	0.365	S	Bot Proj. Player Dist.	0.119	0.133	0.149	0.000
S	Bot Aim (Player)	0.158	0.151	0.149	0.195	S	Bot Shooting	0.113	0.146	0.171	0.053
G	Event Diversity	0.151	0.128	0.150	0.243	S	Player Damaged	0.089	0.112	0.146	0.060
S	Bot Shooting	0.148	0.071	0.170	0.195	S	Bot Δ Rot.	0.085	0.123	0.089	0.000
G	Event Intensity	0.144	0.130	0.172	0.180	S	Player Proj. Count	0.082	0.111	0.128	0.022
S	Player Damaged	0.136	0.102	0.109	0.220	S	Bot Aim (Player)	0.081	0.110	0.149	0.053
S	Player Healing	0.126	0.132	0.000	0.203	G	Bot Movement	0.081	0.133	0.060	0.000
S	Player Proj. Count	0.116	0.093	0.177	0.084	S	Player Proj. Dist.	0.080	0.108	0.122	-0.003
S	Player Kill Count	0.113	0.118	0.148	0.134	G	Bot Diversity	0.078	0.145	0.123	0.127
S	Player Proj. Dist.	0.107	0.081	0.170	0.088	G	Event Diversity	0.076	0.083	0.156	0.037
G	Input Diversity	0.095	0.086	0.134	0.075	S	Player Aim (Enemy)	0.075	0.138	0.157	0.009
G	Input Intensity	0.093	0.086	0.157	0.057	S	Player Shooting	0.075	0.059	0.127	0.022
S	Player Health	-0.091	-0.200	-0.223	0.000	S	Bot Speed X	0.071	0.108	0.050	0.000
S	Player Aim (Enemy)	0.085	0.063	0.117	0.115	S	Bot Speed Z	0.069	0.116	0.037	0.000
S	Bot Health	-0.083	-0.102	-0.164	0.103	S	Bot Speed Y	0.065	0.113	0.003	0.000
S	Bot Proj. Player Dist.	0.083	0.030	0.178	0.000	G	Bot Count	0.065	0.158	0.097	0.037
S	Player Reloading	0.081	0.131	0.000	0.070	G	Event Intensity	0.060	0.053	0.151	0.024
S	Bot Damaged	0.078	0.088	0.151	-0.016	S	Bot Damaged	0.059	0.129	0.119	-0.008
S	Objects Destroyed	0.077	0.000	0.148	0.000	G	Input Diversity	0.053	0.044	0.079	0.025
S	Bot Proj. Count	0.077	0.036	0.159	0.000	S	Player Kill Count	0.038	0.089	0.064	0.028
S	Player Shooting	0.076	0.038	0.135	0.057	G	Player Activity	0.037	0.033	0.030	0.032
S	Bot Speed X	-0.069	-0.061	-0.149	0.000	G	Input Intensity	0.034	0.016	0.042	0.032
S	Player Aim (Dest.)	0.066	0.000	0.148	0.000	S	Pickup Dist.	0.034	0.000	0.064	0.000
G	Player Activity	0.065	0.053	0.110	0.065	S	Objects Destroyed	0.033	0.000	0.049	0.000

G	Bot Movement	-0.054	-0.032	-0.149	0.000	S	Bot Health	0.031	0.141	0.061	-0.013
S	Bot Speed Z	-0.054	0.004	-0.164	0.000	S	Player Pickup (Health)	-0.031	0.000	-0.060	0.000
S	Player Death	0.049	0.057	0.062	0.000	S	Bot Reloading	-0.031	0.000	0.000	-0.036
G	Bot Diversity	-0.047	-0.080	-0.090	0.121	S	Pickups Count	0.029	0.000	0.035	0.000
S	Player Speed X	0.041	0.012	0.094	0.000	S	Player Health	0.029	-0.013	0.024	0.000
S	Pickup Dist.	-0.034	0.000	-0.053	0.000	S	Player Aim (Destructible)	0.028	0.000	0.034	0.000
S	Player Shoot On Reload	0.028	0.035	0.000	0.034	G	Object Diversity	0.027	0.000	0.044	0.000
S	Player Speed Y	0.028	-0.024	0.023	0.000	S	Player Reloading	-0.022	0.001	0.000	-0.039
G	Object Diversity	-0.027	0.000	-0.004	0.000	G	Object Intensity	0.021	0.000	0.035	0.000
S	Pickup Count	-0.026	0.000	-0.009	0.000	G	Time Passed	-0.020	-0.054	-0.030	0.022
S	Bot Speed Y	0.025	0.027	-0.004	0.000	S	Player Speed Y	0.017	-0.002	-0.036	0.000
S	Player Speed Z	0.018	0.026	0.015	0.000	S	Destructible Count	0.016	0.000	0.006	0.000
S	Bot Δ Rot.	-0.018	-0.008	-0.028	0.000	S	Player Healing	-0.015	-0.027	0.000	0.019
S	Bot Reloading	0.016	0.000	0.000	0.095	G	Player Score	-0.015	-0.049	-0.029	0.040
S	Player Crouching	0.016	0.015	0.000	0.000	S	Player Death	-0.014	-0.035	-0.002	0.000
G	Bot Count	0.014	-0.085	-0.118	0.409	S	Player Speed X	0.013	0.046	-0.052	0.000
S	Player Δ Rot.	-0.012	0.008	-0.059	0.014	G	Player Movement	-0.013	0.000	-0.025	0.004
S	Destructible Count	0.009	0.000	0.129	0.000	S	Player Speed Z	-0.011	-0.027	-0.099	0.000
G	Player Movement	0.004	-0.004	-0.012	0.039	S	Player Δ Rot.	0.006	-0.003	0.048	-0.004
S	Player Pickup (Health)	0.003	0.000	0.013	0.000	S	Reticle Δ Dist.	0.006	0.004	0.029	0.004
G	Object Intensity	-0.003	0.000	0.066	0.000	S	Player Sprinting	-0.004	-0.015	0.000	0.000
S	Player Sprinting	-0.002	-0.014	0.000	0.000	S	Player Crouching	-0.004	-0.019	0.000	0.000
S	Reticle Δ Dist.	-0.001	0.000	-0.030	0.039	S	Player Shoot On Reload	0.001	0.018	0.000	-0.012

Table A.6: Kendall's τ correlation between platformer features and arousal.

μA						∇A					
	Feature	**Platformer Genre**	**Endless**	**Pirates!**	**Run'N'Gun**		**Feature**	**Platformer Genre**	**Endless**	**Pirates!**	**Run'N'Gun**
G	Time Passed	0.271	0.247	0.219	0.346	S	Player Health	0.120	0.150	0.098	0.065
G	Player Score	0.269	0.262	0.211	0.328	G	Bot Movement	0.099	0.124	0.077	0.010
S	Bot Proj. Count	0.114	0.000	0.000	0.234	S	Player Pickup (Boost)	0.097	0.108	0.000	0.000
S	Bot Proj. Player Dist.	0.113	0.000	0.000	0.228	G	Bot Diversity	0.079	0.098	0.069	0.075
G	Bot Diversity	0.106	0.062	0.066	0.223	G	Bot Count	0.074	0.084	0.063	0.077
G	Event Intensity	0.087	0.068	0.060	0.110	S	Player Speed Y	-0.069	-0.067	-0.022	0.008
G	Event Diversity	0.083	0.070	0.052	0.104	S	Player Collision Above	0.065	0.027	0.018	-0.008
G	Bot Count	0.081	0.050	0.083	0.132	S	Bot Player Dist.	0.063	0.094	0.048	0.026
S	Bot Speed X	0.077	0.210	0.059	-0.028	S	Bot Health	0.062	0.088	0.071	0.017
G	Player Movement	0.076	0.291	0.061	0.007	S	Bot Collision Below	0.062	0.100	0.059	0.010
S	Player Health	-0.073	-0.005	-0.014	-0.203	S	Bot Grounded	0.062	0.100	0.059	0.010
S	Player on Power	0.069	0.000	0.078	0.000	S	Player Pickup (Point)	0.060	0.066	-0.020	0.000
S	Player Speed X	0.066	0.289	0.012	-0.054	S	Player Falling	-0.059	-0.022	-0.027	0.025
S	Bot Health	0.062	0.062	0.065	0.069	S	Player Coll.	0.055	0.065	-0.019	-0.021
S	Player Pickup (Point)	0.062	0.094	0.062	0.000	S	Player Speed X	0.053	0.082	0.058	-0.011
S	Bot Coll.	0.059	0.000	0.054	0.103	S	Bot Speed X	0.051	0.124	0.062	0.012
S	Bot Player Dist.	0.058	0.047	0.059	0.097	S	Bot Coll.	-0.051	0.000	0.060	0.010
S	Bot Collision Below	0.058	0.071	0.054	0.103	S	Player Jumping	-0.050	0.000	0.055	0.014
S	Bot Grounded	0.058	0.071	0.054	0.103	S	Player Kill Count	0.047	0.095	0.043	-0.007
S	Bot Shooting	0.054	0.000	0.000	0.155	G	Event Diversity	0.041	0.040	0.046	0.043
S	Player Speed Y	0.053	0.046	0.073	0.034	S	Pickup Dist.	0.041	0.085	-0.004	-0.013
S	Player Damaged	0.051	0.034	0.015	0.105	S	Player Death	-0.040	-0.103	-0.032	-0.004
S	Player Falling	0.049	0.044	0.056	0.024	S	Player Proj. Dist.	-0.039	0.000	0.000	0.019
G	Object Intensity	0.047	0.031	0.103	0.036	S	Player Shooting	0.039	0.049	0.000	0.052

S	Pickup Count	0.047	0.031	0.103	0.036	S	Player Collision Right	-0.038	0.000	0.002	0.001
G	Object Diversity	0.046	0.046	0.075	0.032	S	Player Proj. Count	-0.036	0.000	0.000	0.038
S	Player Coll.	-0.044	-0.023	-0.056	-0.021	S	Bot Damaged	-0.035	0.000	0.000	0.002
S	Player Jumping	0.041	0.000	0.024	0.016	S	Player on Power	-0.034	0.000	-0.028	0.000
S	Pickup Dist.	0.039	0.025	0.099	0.006	S	Pickups Count	0.029	0.096	0.002	-0.007
S	Player Kill Count	0.037	0.078	0.038	0.032	G	Object Intensity	0.029	0.096	0.002	-0.007
G	Bot Movement	0.036	0.210	0.036	-0.045	S	Bot Speed Y	-0.028	0.000	0.062	0.027
S	Player Collision Below	-0.035	-0.019	-0.058	-0.024	G	Player Movement	0.028	0.078	-0.029	-0.010
S	Player Grounded	-0.035	-0.019	-0.058	-0.024	G	Player Activity	-0.024	0.028	0.056	-0.002
S	Player Death	0.034	0.036	0.046	0.043	S	Player Collision Left	-0.023	0.000	-0.020	0.010
S	Bot Collision Left	0.031	0.036	-0.024	0.092	S	Bot Falling	-0.023	0.000	0.060	0.026
S	Player Pickup (Power)	0.023	0.000	0.019	0.000	S	Bot Charging	-0.022	0.000	0.000	0.027
S	Player Pickup (Boost)	0.022	0.062	0.000	0.000	S	Player Pickup (Health)	-0.021	0.000	0.000	-0.010
S	Bot Falling	0.021	0.000	0.056	-0.039	S	Bot Shooting	-0.021	0.000	0.000	0.045
S	Bot Collision Right	0.021	0.000	-0.021	0.069	S	Bot Collision Left	-0.018	-0.101	0.031	0.007
S	Bot Speed Y	0.020	0.000	0.055	-0.037	S	Player Pickup (Slow)	0.017	-0.016	0.000	0.000
S	Player Collision Left	0.019	0.000	0.029	0.004	G	Event Intensity	0.016	0.021	0.007	0.030
S	Player Pickup (Slow)	0.018	0.045	0.000	0.000	S	Player Damaged	-0.014	-0.101	0.031	0.042
S	Bot Collision Above	0.017	0.000	0.000	0.036	G	Time Passed	0.010	0.025	-0.008	0.013
G	Input Diversity	0.016	0.037	0.005	-0.036	G	Object Diversity	0.010	0.096	-0.015	-0.004
G	Input Intensity	0.015	0.037	0.003	-0.027	G	Player Score	0.008	0.030	-0.006	0.007
G	Player Activity	0.010	0.032	-0.013	-0.017	S	Player Collision Below	-0.008	0.047	-0.017	-0.020
S	Bot Jumping	0.008	0.000	0.000	0.015	S	Player Grounded	-0.008	0.047	-0.017	-0.020
S	Player Pickup (Health)	-0.007	0.000	0.000	-0.004	S	Bot Proj. Player Dist.	-0.007	0.000	0.000	0.042
S	Bot Charging	0.006	0.000	0.000	0.027	G	Input Intensity	-0.005	0.015	0.058	0.014
S	Player Proj. Count	0.005	0.000	0.000	0.047	S	Bot Collision Above	-0.004	0.000	0.000	0.021
S	Player Shooting	-0.004	0.015	0.000	0.053	S	Bot Proj. Count	-0.004	0.000	0.000	0.052
S	Player Collision Above	0.004	0.032	-0.025	0.015	S	Player Pickup (Power)	-0.003	0.000	0.013	0.000
S	Bot Damaged	0.002	0.000	0.000	0.035	G	Input Diversity	-0.002	0.014	0.074	0.019
S	Player Proj. Dist.	0.002	0.000	0.000	0.027	S	Bot Jumping	-0.002	0.000	0.000	-0.002
S	Player Collision Right	-0.001	0.000	-0.020	-0.022	S	Bot Collision Right	-0.002	0.000	0.030	-0.004

A.3 Feature Importance

This section details the results of the feature importance analysis presented in the thesis.

A.3.1 Feature Importance in Game-Based Models

Tables A.7, A.8, and A.9 show μA and ∇A models for racing, shooter, and platformer games respectively. Models are trained on a combined feature set of general and genre-specific features. Rows marked with 'G' indicate general features, and rows marked with 'S' indicate genre-specific features throughout the tables in this section. The tables are sorted by the best average values per genre.

Table A.7: MDI feature importance in racing game models trained on one game.

μA						∇A					
	Feature	**Racing Average**	**TinyCars**	**Solid**	**ApexSpeed**		**Feature**	**Racing Average**	**TinyCars**	**Solid**	**ApexSpeed**
G	Time Passed	0.089	0.070	0.110	0.089	S	Player Coll.	0.036	0.041	0.023	0.044
G	Player Score	0.085	0.068	0.099	0.090	G	Time Passed	0.036	0.042	0.025	0.041
S	Player Gas Pedal	0.062	0.052	0.040	0.093	G	Player Score	0.035	0.040	0.023	0.041
G	Player Activity	0.045	0.056	0.068	0.011	S	Player Speed	0.035	0.044	0.028	0.032
S	Bot Score	0.033	0.048	0.028	0.023	S	Player Dist. To WP	0.033	0.035	0.029	0.037
S	Player Standing	0.032	0.036	0.023	0.035	G	Player Movement	0.032	0.024	0.040	0.032
S	Bot Coll.	0.031	0.033	0.040	0.021	S	Bot Coll.	0.031	0.047	0.018	0.029
S	Player Lap	0.027	0.012	0.030	0.039	G	Object Intensity	0.028	0.023	0.039	0.023
S	Bot Grounded	0.026	0.024	0.018	0.038	S	Player Steering	0.028	0.025	0.021	0.037
S	Player Respawn	0.026	0.018	0.032	0.028	S	Bot Steering	0.028	0.027	0.030	0.025
S	Bot Player Dist.	0.026	0.021	0.018	0.039	G	Object Diversity	0.027	0.023	0.033	0.025
G	Bot Count	0.025	0.027	0.020	0.028	S	Player Speed Boost	0.026	0.000	0.078	0.000
S	Bot Steering	0.025	0.029	0.014	0.032	S	Player Delta Rot.	0.026	0.024	0.019	0.034
S	Bot Dist. To WP	0.024	0.021	0.023	0.030	G	Bot Movement	0.025	0.025	0.032	0.019
G	Bot Movement	0.023	0.024	0.016	0.030	S	Player Standing	0.025	0.028	0.025	0.022

S	Bot Lap	0.022	0.020	0.025	0.021	S	Bot Gas Pedal	0.025	0.027	0.026	0.021
S	Bot Standing	0.022	0.027	0.019	0.020	S	Bot Player Dist.	0.025	0.027	0.021	0.027
S	Bot Delta Rot.	0.022	0.017	0.016	0.032	G	Bot Count	0.025	0.024	0.029	0.021
G	Bot Diversity	0.021	0.022	0.019	0.022	G	Event Intensity	0.025	0.030	0.021	0.022
S	Player Speed	0.020	0.030	0.015	0.016	G	Player Activity	0.024	0.027	0.022	0.023
S	Player Coll.	0.020	0.026	0.017	0.016	S	Bot Delta Rot.	0.024	0.024	0.027	0.021
G	Event Intensity	0.019	0.017	0.024	0.017	S	Bot Speed	0.023	0.030	0.021	0.019
S	Bot Gas Pedal	0.019	0.023	0.013	0.021	S	Bot Dist. To WP	0.023	0.024	0.026	0.020
S	Bot Speed	0.019	0.024	0.016	0.017	G	Event Diversity	0.023	0.026	0.022	0.020
G	Event Diversity	0.018	0.017	0.023	0.016	S	Player Off Road	0.022	0.038	0.000	0.028
S	Player Grounded	0.018	0.020	0.021	0.014	S	Bot Grounded	0.022	0.021	0.025	0.020
S	Player Mid Air	0.018	0.019	0.021	0.014	G	Input Intensity	0.021	0.023	0.019	0.022
G	Player Movement	0.016	0.026	0.013	0.010	G	Input Diversity	0.021	0.023	0.020	0.022
S	Player Delta Rot.	0.015	0.018	0.016	0.010	G	Bot Diversity	0.021	0.018	0.028	0.018
G	Input Diversity	0.015	0.019	0.014	0.011	S	Bot Standing	0.021	0.025	0.019	0.020
S	Player Dist. To WP	0.013	0.017	0.012	0.010	S	Player Mid Air	0.021	0.021	0.022	0.020
G	Input Intensity	0.013	0.017	0.013	0.008	S	Player Gas Pedal	0.021	0.029	0.014	0.020
S	Bot Off Road	0.013	0.014	0.024	0.000	S	Player Grounded	0.021	0.021	0.022	0.019
S	Player Off Road	0.012	0.022	0.015	0.000	S	Player Looping	0.020	0.000	0.000	0.060
G	Object Intensity	0.012	0.018	0.010	0.008	S	Bot Score	0.019	0.025	0.014	0.018
G	Object Diversity	0.011	0.018	0.009	0.008	S	Jump Count	0.018	0.023	0.030	0.000
S	Player Steering	0.010	0.013	0.009	0.009	S	Player Lap	0.015	0.010	0.014	0.020
S	Player Looping	0.010	0.000	0.029	0.000	S	Player Respawn	0.011	0.011	0.014	0.007
S	Jump Count	0.009	0.018	0.000	0.009	S	Obstacle Count	0.011	0.000	0.033	0.000
S	Player Speed Boost	0.006	0.000	0.000	0.018	S	Speed Boost Count	0.009	0.000	0.028	0.000
S	Bot Looping	0.006	0.000	0.017	0.000	S	Bot Lap	0.009	0.010	0.006	0.012
S	Speed Boost Count	0.006	0.000	0.000	0.017	S	Bot Off Road	0.009	0.010	0.000	0.018
S	Bot Speed Boost	0.005	0.000	0.000	0.016	S	Loop Count	0.007	0.000	0.000	0.022
S	Loop Count	0.003	0.000	0.010	0.000	S	Bot Looping	0.007	0.000	0.000	0.020
S	Obstacle Count	0.003	0.000	0.000	0.009	S	Bot Speed Boost	0.004	0.000	0.011	0.000
S	Bot Respawn	0.003	0.000	0.000	0.009	S	Bot Respawn	0.001	0.001	0.003	0.000

Table A.8: MDI feature importance in shooter game models trained on one game.

μA						∇A					
	Feature	**Shooter Average**	**Heist!**	**TopDown**	**Shootout**		**Feature**	**Shooter Average**	**Heist!**	**TopDown**	**Shootout**
G	Time Passed	0.167	0.165	0.187	0.148	S	Bot Shooting	0.056	0.053	0.068	0.045
G	Player Score	0.126	0.140	0.164	0.072	S	Bot Health	0.046	0.065	0.039	0.034
S	Bot Health	0.054	0.017	0.012	0.132	G	Bot Diversity	0.046	0.069	0.037	0.031
G	Bot Count	0.051	0.016	0.008	0.128	G	Bot Count	0.043	0.062	0.027	0.042
G	Bot Diversity	0.050	0.016	0.012	0.122	S	Player Damaged	0.042	0.028	0.044	0.055
S	Bot Reloading	0.042	0.000	0.000	0.125	S	Player Aim (Enemy)	0.041	0.034	0.048	0.041
S	Player Health	0.036	0.062	0.044	0.000	G	Event Diversity	0.035	0.018	0.049	0.038
G	Event Diversity	0.026	0.028	0.017	0.032	S	Bot Aim (Player)	0.035	0.026	0.034	0.044
G	Player Activity	0.024	0.009	0.053	0.010	S	Bot Proj. Count	0.031	0.044	0.048	0.000
G	Event Intensity	0.023	0.024	0.019	0.027	S	Bot Damaged	0.029	0.026	0.021	0.039
S	Player Pickup (Health)	0.022	0.000	0.066	0.000	G	Event Intensity	0.028	0.016	0.030	0.038
S	Player Proj. Count	0.021	0.020	0.033	0.011	S	Player Proj. Dist.	0.027	0.021	0.018	0.043
S	Player Reloading	0.019	0.038	0.000	0.020	G	Time Passed	0.027	0.023	0.018	0.040
S	Player Damaged	0.018	0.021	0.015	0.016	S	Player Kill Count	0.027	0.021	0.024	0.036
S	Player Healing	0.017	0.037	0.000	0.015	S	Player Proj. Count	0.027	0.022	0.019	0.040
S	Bot Shooting	0.017	0.019	0.021	0.011	S	Player Shooting	0.027	0.017	0.025	0.038
S	Bot Damaged	0.016	0.022	0.015	0.012	S	Bot Proj. Player Dist.	0.026	0.039	0.041	0.000
S	Bot Aim (Player)	0.016	0.027	0.012	0.010	G	Input Intensity	0.026	0.023	0.016	0.038
S	Player Kill Count	0.016	0.021	0.014	0.014	G	Input Diversity	0.026	0.023	0.016	0.038
S	Bot Speed Y	0.016	0.039	0.009	0.000	G	Player Activity	0.026	0.020	0.019	0.037
S	Player Proj. Dist.	0.016	0.013	0.021	0.014	S	Player Reloading	0.025	0.019	0.000	0.056
S	Player Shooting	0.015	0.015	0.011	0.019	S	Reticle Delta Dist.	0.025	0.017	0.021	0.036
S	Bot Delta Rot.	0.015	0.037	0.008	0.000	G	Player Movement	0.024	0.017	0.018	0.036
G	Input Intensity	0.014	0.008	0.023	0.010	G	Player Score	0.024	0.019	0.016	0.036
S	Player Aim (Enemy)	0.014	0.017	0.010	0.014	S	Player Delta Rot.	0.024	0.018	0.018	0.035

S	Player Speed Y	0.013	0.023	0.016	0.000	G	Bot Movement	0.020	0.037	0.022	0.000
S	Bot Speed Z	0.013	0.021	0.018	0.000	S	Player Healing	0.017	0.016	0.000	0.036
S	Bot Speed X	0.013	0.022	0.017	0.000	S	Bot Speed Z	0.017	0.030	0.021	0.000
S	Bot Proj. Player Dist.	0.011	0.013	0.020	0.000	S	Bot Speed X	0.016	0.028	0.019	0.000
G	Input Diversity	0.009	0.009	0.009	0.010	S	Bot Delta Rot.	0.015	0.020	0.024	0.000
S	Player Delta Rot.	0.009	0.007	0.011	0.009	S	Player Health	0.014	0.020	0.021	0.000
S	Bot Proj. Count	0.009	0.015	0.012	0.000	S	Bot Reloading	0.012	0.000	0.000	0.037
G	Bot Movement	0.008	0.012	0.011	0.000	S	Player Speed X	0.012	0.019	0.018	0.000
S	Reticle Delta Dist.	0.008	0.007	0.007	0.009	S	Player Speed Z	0.011	0.017	0.016	0.000
G	Player Movement	0.007	0.007	0.007	0.009	S	Player Speed Y	0.009	0.020	0.008	0.000
S	Player Death	0.007	0.013	0.009	0.000	S	Bot Speed Y	0.007	0.019	0.003	0.000
S	Player Speed Z	0.007	0.012	0.008	0.000	S	Player Death	0.007	0.014	0.008	0.000
S	Player Speed X	0.005	0.008	0.008	0.000	S	Pickup Dist.	0.006	0.000	0.018	0.000
S	Objects Destroyed	0.004	0.000	0.013	0.000	S	Player Aim (Destructible)	0.006	0.000	0.018	0.000
S	Pickup Dist.	0.004	0.000	0.012	0.000	G	Object Intensity	0.005	0.000	0.016	0.000
S	Destructible Count	0.004	0.000	0.011	0.000	S	Player Pickup (Health)	0.005	0.000	0.016	0.000
G	Object Diversity	0.003	0.000	0.010	0.000	S	Destructible Count	0.005	0.000	0.016	0.000
S	Player Aim (Destructible)	0.003	0.000	0.009	0.000	G	Object Diversity	0.005	0.000	0.015	0.000
S	Player Crouching	0.003	0.009	0.000	0.000	S	Objects Destroyed	0.005	0.000	0.015	0.000
G	Object Intensity	0.003	0.000	0.009	0.000	S	Pickups	0.005	0.000	0.015	0.000
S	Pickups	0.003	0.000	0.008	0.000	S	Player Shoot On Reload	0.004	0.002	0.000	0.010
S	Player Sprinting	0.002	0.006	0.000	0.000	S	Player Crouching	0.003	0.010	0.000	0.000
S	Player Shoot On Reload	0.001	0.003	0.000	0.001	S	Player Sprinting	0.002	0.007	0.000	0.000

Table A.9: MDI feature importance in platformer game models trained on one game.

μA						∇A					
Feature		**Platformer Average**	**Endless**	**Pirates!**	**Run'N'Gun**	**Feature**		**Platformer Average**	**Endless**	**Pirates!**	**Run'N'Gun**
G	Time Passed	0.106	0.145	0.092	0.082	G	Bot Movement	0.043	0.062	0.044	0.024
G	Player Score	0.104	0.129	0.088	0.095	S	Bot Speed X	0.038	0.068	0.024	0.021
S	Player Damaged	0.039	0.013	0.065	0.038	G	Bot Count	0.037	0.019	0.036	0.057
G	Bot Movement	0.037	0.047	0.022	0.041	S	Player Health	0.036	0.055	0.032	0.020
G	Player Movement	0.035	0.007	0.014	0.083	S	Player Speed X	0.031	0.046	0.029	0.018
S	Player Speed X	0.034	0.008	0.012	0.082	G	Bot Diversity	0.029	0.022	0.029	0.035
S	Player Health	0.031	0.031	0.035	0.028	S	Bot Health	0.027	0.037	0.026	0.020
S	Player Death	0.029	0.005	0.044	0.038	G	Player Movement	0.027	0.039	0.023	0.019
S	Player Coll. Above	0.026	0.041	0.024	0.013	S	Bot Player Dist.	0.026	0.030	0.028	0.020
G	Bot Diversity	0.025	0.042	0.019	0.015	G	Event Diversity	0.025	0.017	0.021	0.038
S	Bot Coll. Left	0.025	0.009	0.031	0.036	S	Bot Coll. Below	0.025	0.034	0.022	0.018
S	Player Pickup (Health)	0.023	0.070	0.000	0.000	S	Bot Grounded	0.025	0.033	0.022	0.019
S	Bot Speed X	0.022	0.011	0.016	0.040	G	Object Diversity	0.024	0.033	0.022	0.018
S	Player Coll. Left	0.021	0.035	0.028	0.000	S	Pickup Dist.	0.024	0.032	0.021	0.019
S	Player Kill Count	0.018	0.010	0.027	0.017	S	Player Damaged	0.023	0.029	0.018	0.022
S	Player Coll. Right	0.017	0.025	0.025	0.000	S	Player Speed Y	0.023	0.024	0.027	0.017
G	Object Intensity	0.016	0.008	0.025	0.015	G	Time Passed	0.023	0.019	0.025	0.023
G	Bot Count	0.016	0.010	0.019	0.017	G	Input Intensity	0.022	0.018	0.026	0.023
S	Bot Speed Y	0.015	0.025	0.022	0.000	S	Pickups	0.022	0.026	0.020	0.020
S	Pickups	0.015	0.008	0.024	0.014	G	Object Intensity	0.022	0.026	0.020	0.019
S	Player Pickup (Power)	0.015	0.000	0.046	0.000	S	Player Falling	0.022	0.021	0.024	0.020
S	Pickup Dist.	0.015	0.010	0.018	0.017	G	Player Activity	0.021	0.018	0.027	0.020
S	Bot Health	0.015	0.011	0.016	0.017	S	Player Kill Count	0.021	0.021	0.022	0.021
S	Player Pickup (Point)	0.014	0.000	0.022	0.021	S	Player Coll.	0.021	0.025	0.020	0.019

S	Player Speed Y	0.014	0.007	0.013	0.022	G	Input Diversity	0.021	0.017	0.026	0.019
G	Object Diversity	0.014	0.008	0.019	0.015	G	Player Score	0.021	0.019	0.021	0.021
S	Bot Player Dist.	0.014	0.012	0.012	0.018	G	Event Intensity	0.020	0.017	0.018	0.025
S	Bot Coll. Below	0.014	0.011	0.013	0.017	S	Player Coll. Above	0.019	0.017	0.023	0.015
S	Bot Grounded	0.014	0.010	0.014	0.017	S	Bot Coll. Left	0.018	0.022	0.023	0.008
G	Event Diversity	0.013	0.014	0.013	0.013	S	Player Coll. Below	0.017	0.017	0.018	0.017
S	Player Coll.	0.013	0.006	0.011	0.021	S	Player Grounded	0.017	0.017	0.018	0.016
G	Player Activity	0.013	0.010	0.012	0.016	S	Player Pickup (Point)	0.016	0.027	0.021	0.000
G	Input Intensity	0.012	0.009	0.012	0.016	S	Player Death	0.016	0.025	0.019	0.003
G	Event Intensity	0.012	0.011	0.014	0.012	S	Bot Falling	0.016	0.000	0.028	0.019
S	Player Falling	0.012	0.007	0.012	0.016	S	Player Coll. Right	0.015	0.000	0.023	0.021
S	Bot Coll. Right	0.012	0.009	0.026	0.000	S	Bot Speed Y	0.014	0.000	0.023	0.020
G	Input Diversity	0.012	0.009	0.011	0.015	S	Player Jumping	0.014	0.000	0.025	0.018
S	Bot Falling	0.010	0.013	0.017	0.000	S	Bot Coll.	0.014	0.000	0.023	0.019
S	Player Coll. Below	0.010	0.006	0.011	0.013	S	Player Shooting	0.014	0.019	0.000	0.022
S	Player Grounded	0.010	0.006	0.011	0.013	S	Bot Shooting	0.012	0.000	0.000	0.035
S	Player Shooting	0.009	0.009	0.000	0.016	S	Player Coll. Left	0.012	0.000	0.019	0.016
S	Player Pickup (Boost)	0.008	0.000	0.000	0.025	S	Player Pickup (Boost)	0.010	0.030	0.000	0.000
S	Bot Coll.	0.008	0.011	0.014	0.000	S	Bot Coll. Right	0.009	0.000	0.021	0.005
S	Bot Coll. Above	0.008	0.024	0.000	0.000	S	Bot Damaged	0.008	0.000	0.000	0.024
S	Player Pickup (Slow)	0.008	0.000	0.000	0.023	S	Player Pickup (Power)	0.008	0.000	0.023	0.000
S	Bot Proj. Player Dist.	0.007	0.022	0.000	0.000	S	Bot Charging	0.007	0.000	0.000	0.022
S	Player on Power	0.007	0.000	0.021	0.000	S	Player on Power	0.007	0.000	0.021	0.000
S	Bot Shooting	0.006	0.018	0.000	0.000	S	Bot Proj. Count	0.007	0.000	0.000	0.020
S	Player Jumping	0.006	0.007	0.011	0.000	S	Player Proj. Dist.	0.006	0.000	0.000	0.019
S	Bot Proj. Count	0.006	0.018	0.000	0.000	S	Player Pickup (Health)	0.006	0.000	0.000	0.018
S	Bot Damaged	0.004	0.012	0.000	0.000	S	Player Pickup (Slow)	0.006	0.018	0.000	0.000
S	Bot Charging	0.004	0.011	0.000	0.000	S	Player Proj. Count	0.006	0.000	0.000	0.018
S	Player Proj. Dist.	0.003	0.009	0.000	0.000	S	Bot Proj. Player Dist.	0.005	0.000	0.000	0.015
S	Player Proj. Count	0.003	0.009	0.000	0.000	S	Bot Coll. Above	0.004	0.000	0.000	0.012
S	Bot Jumping	0.000	0.001	0.000	0.000	S	Bot Jumping	0.000	0.000	0.000	0.001

A.3.2 Feature Importance in Genre-Based Models for Unseen Games

Tables A.10, A.11 and A.12 show μA and ∇A models for racing, shooter, and platformer genres respectively. Models are trained on a combined feature set of general and genre-specific features. Rows marked with 'G' indicate general features, and rows marked with 'S' indicate genre-specific features throughout the tables in this section. The tables are sorted by the best average values per genre.

Table A.10: MDI feature importance in racing game models trained on 2 games.

μA						∇A					
	Feature	**Racing Average**	**TinyCars & Solid**	**TinyCars & ApexSpeed**	**Solid & ApexSpeed**		**Feature**	**Racing Average**	**TinyCars & Solid**	**TinyCars & ApexSpeed**	**Solid & ApexSpeed**
G	Time Passed	0.116	0.107	0.112	0.130	G	Time Passed	0.046	0.050	0.045	0.043
G	Player Score	0.110	0.100	0.108	0.123	S	Player Coll.	0.043	0.051	0.036	0.042
S	Player Gas Pedal	0.066	0.047	0.080	0.071	G	Player Score	0.042	0.046	0.038	0.040
G	Player Activity	0.038	0.063	0.024	0.029	S	Player Speed	0.038	0.041	0.042	0.031
S	Bot Coll.	0.038	0.041	0.034	0.038	G	Player Movement	0.033	0.027	0.031	0.042
S	Bot Score	0.033	0.039	0.032	0.027	S	Bot Coll.	0.033	0.040	0.034	0.025
S	Player Lap	0.032	0.021	0.032	0.045	S	Player Speed Boost	0.032	0.000	0.050	0.045
S	Player Standing	0.026	0.029	0.028	0.022	G	Object Intensity	0.029	0.022	0.036	0.029
G	Bot Count	0.024	0.022	0.024	0.027	S	Player Dist. To WP	0.029	0.038	0.025	0.024
S	Player Speed	0.024	0.022	0.030	0.018	G	Object Diversity	0.028	0.023	0.032	0.031
S	Bot Dist. To WP	0.024	0.018	0.020	0.032	G	Bot Count	0.027	0.023	0.029	0.027
S	Bot Grounded	0.023	0.021	0.024	0.023	S	Bot Gas Pedal	0.025	0.027	0.024	0.024
S	Bot Steering	0.022	0.017	0.034	0.015	S	Player Steering	0.024	0.024	0.020	0.029
S	Bot Lap	0.021	0.024	0.019	0.021	S	Player Delta Rot.	0.024	0.028	0.019	0.025
S	Bot Standing	0.021	0.023	0.023	0.017	S	Player Standing	0.024	0.024	0.026	0.022
S	Player Coll.	0.020	0.021	0.021	0.018	S	Bot Grounded	0.024	0.023	0.025	0.024
S	Player Respawn	0.019	0.023	0.019	0.016	G	Bot Diversity	0.024	0.019	0.027	0.024
G	Bot Movement	0.019	0.019	0.023	0.017	S	Bot Player Dist.	0.022	0.026	0.020	0.019
S	Bot Player Dist.	0.019	0.020	0.018	0.017	S	Bot Delta Rot.	0.022	0.020	0.023	0.022

G	Bot Diversity	0.019	0.018	0.018	0.019	G	Event Intensity	0.021	0.024	0.022	0.019
G	Event Intensity	0.018	0.016	0.015	0.022	S	Bot Steering	0.021	0.023	0.021	0.020
S	Bot Speed	0.017	0.019	0.019	0.013	S	Bot Speed	0.021	0.022	0.023	0.019
S	Bot Delta Rot.	0.017	0.015	0.018	0.018	G	Bot Movement	0.021	0.021	0.022	0.021
G	Event Diversity	0.017	0.016	0.014	0.020	G	Player Activity	0.021	0.024	0.021	0.019
S	Player Mid Air	0.017	0.019	0.016	0.015	S	Player Off Road	0.021	0.032	0.017	0.014
S	Player Grounded	0.016	0.018	0.016	0.015	S	Jump Count	0.021	0.012	0.028	0.022
S	Bot Gas Pedal	0.016	0.017	0.018	0.013	S	Player Looping	0.021	0.034	0.000	0.029
G	Player Movement	0.015	0.021	0.014	0.010	S	Bot Dist. To WP	0.021	0.019	0.024	0.019
S	Player Delta Rot.	0.013	0.018	0.011	0.011	G	Event Diversity	0.021	0.022	0.021	0.019
G	Input Diversity	0.012	0.015	0.012	0.010	S	Bot Standing	0.020	0.021	0.021	0.019
S	Player Off Road	0.012	0.015	0.013	0.009	S	Player Gas Pedal	0.020	0.021	0.021	0.016
S	Bot Off Road	0.011	0.016	0.006	0.011	G	Input Diversity	0.019	0.021	0.019	0.018
G	Input Intensity	0.011	0.013	0.010	0.008	G	Input Intensity	0.019	0.020	0.018	0.018
S	Player Dist. To WP	0.010	0.012	0.011	0.009	S	Player Grounded	0.018	0.017	0.019	0.019
G	Object Intensity	0.010	0.011	0.012	0.007	S	Player Mid Air	0.018	0.017	0.019	0.019
S	Player Looping	0.010	0.015	0.000	0.014	S	Bot Score	0.017	0.019	0.018	0.015
S	Jump Count	0.010	0.013	0.011	0.005	S	Player Lap	0.016	0.017	0.012	0.019
G	Object Diversity	0.009	0.011	0.011	0.007	S	Obstacle Count	0.012	0.000	0.020	0.017
S	Player Steering	0.009	0.009	0.009	0.007	S	Speed Boost Count	0.012	0.000	0.018	0.018
S	Speed Boost Count	0.007	0.000	0.011	0.009	S	Player Respawn	0.010	0.009	0.012	0.009
S	Player Speed Boost	0.006	0.000	0.010	0.008	S	Loop Count	0.009	0.016	0.000	0.012
S	Bot Looping	0.006	0.008	0.000	0.009	S	Bot Lap	0.009	0.010	0.007	0.008
S	Bot Speed Boost	0.006	0.000	0.010	0.007	S	Bot Off Road	0.008	0.013	0.004	0.008
S	Loop Count	0.004	0.008	0.000	0.006	S	Bot Looping	0.006	0.010	0.000	0.008
S	Obstacle Count	0.004	0.000	0.007	0.005	S	Bot Speed Boost	0.004	0.000	0.007	0.007
S	Bot Respawn	0.003	0.000	0.004	0.004	S	Bot Respawn	0.001	0.000	0.002	0.002

Table A.11: MDI feature importance in shooter game models trained on 2 games.

μA						∇A					
	Feature	**Shooter Average**	**Heist! & TopDown**	**Heist! & Shootout**	**TopDown & Shootout**		**Feature**	**Shooter Average**	**Heist! & TopDown**	**Heist! & Shootout**	**TopDown & Shootout**
G	Time Passed	0.225	0.228	0.222	0.225	S	Bot Shooting	0.064	0.073	0.055	0.063
G	Player Score	0.162	0.179	0.150	0.157	G	Bot Diversity	0.052	0.057	0.061	0.038
S	Bot Reloading	0.042	0.000	0.069	0.056	S	Bot Proj. Count	0.049	0.059	0.043	0.045
S	Player Health	0.040	0.063	0.032	0.026	S	Bot Health	0.048	0.049	0.057	0.039
G	Bot Diversity	0.037	0.012	0.035	0.064	S	Player Damaged	0.043	0.040	0.036	0.053
G	Event Diversity	0.029	0.020	0.042	0.024	S	Bot Proj. Player Dist.	0.042	0.048	0.040	0.039
S	Bot Health	0.027	0.013	0.027	0.041	S	Player Aim (Enemy)	0.041	0.051	0.032	0.039
S	Player Pickup (Health)	0.023	0.027	0.000	0.043	G	Bot Count	0.037	0.039	0.043	0.029
G	Event Intensity	0.023	0.024	0.025	0.021	S	Bot Aim (Player)	0.036	0.037	0.034	0.037
S	Bot Speed X	0.021	0.025	0.020	0.019	G	Bot Movement	0.029	0.027	0.038	0.022
S	Player Proj. Count	0.018	0.027	0.013	0.015	G	Event Diversity	0.028	0.028	0.019	0.035
S	Player Damaged	0.018	0.020	0.019	0.015	S	Bot Speed Z	0.023	0.021	0.031	0.018
S	Bot Shooting	0.017	0.020	0.014	0.018	S	Bot Damaged	0.022	0.024	0.022	0.021
S	Player Reloading	0.017	0.013	0.030	0.008	S	Player Proj. Count	0.022	0.020	0.023	0.023
G	Player Activity	0.017	0.027	0.007	0.017	G	Time Passed	0.022	0.019	0.024	0.022
S	Bot Aim (Player)	0.017	0.022	0.016	0.012	S	Bot Speed X	0.021	0.021	0.027	0.016
G	Bot Count	0.015	0.012	0.019	0.016	S	Player Proj. Dist.	0.021	0.018	0.023	0.022
S	Player Kill Count	0.015	0.020	0.013	0.012	S	Player Kill Count	0.021	0.019	0.022	0.023
S	Bot Speed Y	0.015	0.016	0.023	0.005	S	Bot Delta Rot.	0.021	0.019	0.018	0.025
S	Bot Delta Rot.	0.014	0.010	0.026	0.006	G	Event Intensity	0.020	0.018	0.018	0.026
S	Player Proj. Dist.	0.014	0.015	0.011	0.016	G	Input Diversity	0.020	0.017	0.023	0.019
S	Player Healing	0.014	0.014	0.023	0.004	G	Input Intensity	0.019	0.017	0.022	0.019
S	Bot Speed Z	0.013	0.009	0.014	0.016	G	Player Activity	0.019	0.018	0.021	0.019
S	Bot Damaged	0.013	0.018	0.011	0.010	S	Player Reloading	0.019	0.012	0.028	0.017

S	Bot Proj. Player Dist.	0.012	0.014	0.013	0.010	S	Player Shooting	0.019	0.015	0.018	0.024
S	Bot Proj. Count	0.012	0.010	0.018	0.007	S	Reticle Delta Dist.	0.019	0.016	0.019	0.022
S	Player Shooting	0.011	0.010	0.014	0.010	G	Player Score	0.019	0.017	0.021	0.019
G	Bot Movement	0.011	0.008	0.010	0.014	S	Player Health	0.019	0.020	0.017	0.018
S	Player Aim (Enemy)	0.011	0.012	0.012	0.008	S	Player Delta Rot.	0.018	0.015	0.020	0.020
S	Player Speed Y	0.010	0.010	0.013	0.007	G	Player Movement	0.018	0.015	0.019	0.020
G	Input Intensity	0.009	0.012	0.006	0.009	S	Player Speed X	0.015	0.014	0.016	0.015
S	Player Death	0.007	0.012	0.007	0.003	S	Player Speed Z	0.014	0.014	0.014	0.013
S	Player Speed Z	0.007	0.007	0.009	0.006	S	Player Healing	0.012	0.009	0.019	0.010
G	Input Diversity	0.007	0.007	0.007	0.007	S	Player Speed Y	0.012	0.011	0.019	0.007
S	Player Delta Rot.	0.007	0.008	0.006	0.007	S	Bot Speed Y	0.010	0.012	0.017	0.003
S	Player Speed X	0.006	0.006	0.006	0.006	S	Player Death	0.010	0.012	0.012	0.006
S	Reticle Delta Dist.	0.005	0.005	0.005	0.005	S	Pickup Dist.	0.009	0.011	0.000	0.016
G	Player Movement	0.005	0.005	0.005	0.005	G	Object Intensity	0.007	0.009	0.000	0.013
S	Objects Destroyed	0.005	0.005	0.000	0.010	S	Player Pickup (Health)	0.007	0.008	0.000	0.014
S	Pickup Dist.	0.005	0.006	0.000	0.008	S	Player Aim (Destructible)	0.007	0.008	0.000	0.013
S	Destructible Count	0.004	0.005	0.000	0.008	S	Pickups	0.007	0.008	0.000	0.012
G	Object Diversity	0.004	0.005	0.000	0.006	G	Object Diversity	0.007	0.008	0.000	0.012
G	Object Intensity	0.003	0.004	0.000	0.006	S	Destructible Count	0.007	0.008	0.000	0.012
S	Player Aim (Destructible)	0.003	0.004	0.000	0.006	S	Bot Reloading	0.007	0.000	0.010	0.010
S	Player Crouching	0.003	0.004	0.005	0.000	S	Objects Destroyed	0.006	0.007	0.000	0.012
S	Pickups	0.003	0.004	0.000	0.005	S	Player Crouching	0.005	0.006	0.008	0.000
S	Player Sprinting	0.002	0.003	0.003	0.000	S	Player Sprinting	0.003	0.004	0.006	0.000
S	Player Shoot On Reload	0.001	0.001	0.001	0.000	S	Player Shoot On Reload	0.002	0.001	0.003	0.002

Table A.12: MDI feature importance in platformer game models trained on 2 games.

μA						∇A					
	Feature	**Platformer Average**	**Endless & Pirates!**	**Endless & Run'N'Gun**	**Pirates! & Run'N'Gun**		**Feature**	**Platformer Average**	**Endless & Pirates!**	**Endless & Run'N'Gun**	**Pirates! & Run'N'Gun**
G	Time Passed	0.137	0.107	0.153	0.149	G	Bot Movement	0.053	0.069	0.055	0.036
G	Player Score	0.132	0.114	0.141	0.140	S	Player Health	0.042	0.052	0.042	0.032
S	Player Damaged	0.046	0.066	0.036	0.035	G	Bot Count	0.042	0.023	0.045	0.057
S	Player Death	0.035	0.056	0.025	0.025	G	Bot Diversity	0.033	0.027	0.037	0.036
G	Bot Movement	0.032	0.027	0.025	0.043	S	Bot Speed X	0.030	0.047	0.024	0.020
S	Player Health	0.030	0.041	0.025	0.024	S	Bot Health	0.030	0.036	0.030	0.023
G	Player Movement	0.030	0.042	0.038	0.008	S	Player Speed X	0.028	0.040	0.024	0.020
S	Player Collision Left	0.026	0.011	0.024	0.044	S	Bot Player Dist.	0.027	0.029	0.027	0.024
S	Bot Collision Left	0.025	0.037	0.024	0.015	S	Bot Collision Below	0.024	0.030	0.024	0.019
G	Bot Diversity	0.023	0.013	0.025	0.030	G	Event Diversity	0.024	0.016	0.023	0.034
S	Player Speed X	0.021	0.028	0.028	0.008	S	Bot Grounded	0.024	0.030	0.024	0.019
S	Bot Speed X	0.021	0.035	0.019	0.010	G	Player Movement	0.024	0.025	0.028	0.018
S	Player Pickup (Health)	0.021	0.000	0.037	0.026	S	Player Death	0.022	0.027	0.026	0.013
S	Player Collision Above	0.019	0.015	0.014	0.028	S	Pickup Dist.	0.021	0.024	0.021	0.019
S	Player Collision Right	0.018	0.013	0.011	0.032	G	Time Passed	0.020	0.019	0.018	0.024
S	Bot Speed Y	0.017	0.011	0.013	0.026	G	Object Diversity	0.020	0.022	0.020	0.019
S	Bot Collision Below	0.014	0.015	0.018	0.010	S	Player Damaged	0.020	0.021	0.021	0.019
S	Bot Grounded	0.014	0.014	0.018	0.010	S	Player Speed Y	0.020	0.025	0.016	0.019
S	Player Pickup (Point)	0.013	0.019	0.010	0.011	S	Pickups	0.019	0.021	0.018	0.019
S	Bot Player Dist.	0.013	0.014	0.015	0.010	S	Player Falling	0.019	0.020	0.018	0.019
S	Bot Health	0.013	0.016	0.011	0.011	G	Object Intensity	0.019	0.020	0.018	0.018
G	Bot Count	0.013	0.014	0.010	0.013	S	Player Kill Count	0.018	0.020	0.017	0.018
S	Player Kill Count	0.012	0.015	0.010	0.011	G	Player Activity	0.018	0.019	0.016	0.019
S	Pickup Dist.	0.012	0.014	0.011	0.011	G	Input Intensity	0.018	0.018	0.017	0.018

G	Event Diversity	0.012	0.011	0.011	0.014	G	Player Score	0.018	0.018	0.017	0.019
S	Player Speed Y	0.012	0.017	0.010	0.007	G	Event Intensity	0.018	0.015	0.017	0.021
S	Pickups	0.012	0.015	0.009	0.011	S	Player Coll.	0.018	0.018	0.017	0.018
G	Object Intensity	0.011	0.015	0.009	0.011	G	Input Diversity	0.018	0.017	0.016	0.020
S	Player Pickup (Power)	0.011	0.019	0.000	0.015	S	Bot Collision Left	0.018	0.020	0.019	0.015
S	Bot Collision Right	0.011	0.013	0.005	0.016	S	Bot Falling	0.018	0.017	0.012	0.024
G	Event Intensity	0.011	0.011	0.009	0.013	S	Player Pickup (Point)	0.017	0.022	0.020	0.010
G	Object Diversity	0.011	0.014	0.009	0.010	S	Player Pickup (Boost)	0.017	0.024	0.027	0.000
S	Bot Falling	0.010	0.009	0.008	0.014	S	Bot Speed Y	0.017	0.014	0.013	0.023
G	Player Activity	0.010	0.011	0.010	0.009	S	Player Collision Above	0.017	0.017	0.015	0.018
S	Player Coll.	0.010	0.015	0.009	0.006	S	Bot Shooting	0.015	0.000	0.024	0.021
G	Input Intensity	0.010	0.012	0.009	0.009	S	Player Jumping	0.015	0.015	0.011	0.020
S	Bot Proj. Player Dist.	0.010	0.000	0.016	0.014	S	Bot Coll.	0.015	0.014	0.012	0.019
S	Player Falling	0.009	0.013	0.008	0.007	S	Player Collision Right	0.015	0.014	0.012	0.020
G	Input Diversity	0.009	0.011	0.009	0.008	S	Player Grounded	0.015	0.016	0.014	0.015
S	Bot Collision Above	0.009	0.000	0.015	0.011	S	Player Collision Below	0.015	0.015	0.014	0.015
S	Bot Coll.	0.009	0.009	0.007	0.010	S	Player Shooting	0.014	0.010	0.018	0.013
S	Bot Shooting	0.009	0.000	0.014	0.012	S	Player Collision Left	0.012	0.010	0.009	0.017
S	Bot Proj. Count	0.008	0.000	0.013	0.012	S	Bot Damaged	0.009	0.000	0.013	0.015
S	Player Shooting	0.008	0.009	0.011	0.005	S	Bot Charging	0.009	0.000	0.014	0.013
S	Player Pickup (Slow)	0.008	0.011	0.013	0.000	S	Bot Proj. Count	0.009	0.000	0.013	0.014
S	Player Collision Below	0.008	0.010	0.007	0.006	S	Bot Collision Right	0.009	0.012	0.003	0.012
S	Player Grounded	0.008	0.010	0.007	0.006	S	Player Pickup (Health)	0.008	0.000	0.012	0.012
S	Player Pickup (Boost)	0.007	0.011	0.011	0.000	S	Player Pickup (Power)	0.008	0.012	0.000	0.012
S	Player on Power	0.007	0.011	0.000	0.010	S	Player Proj. Dist.	0.008	0.000	0.010	0.012
S	Player Jumping	0.006	0.007	0.004	0.007	S	Player on Power	0.007	0.012	0.000	0.010
S	Bot Damaged	0.004	0.000	0.007	0.007	S	Player Proj. Count	0.007	0.000	0.010	0.011
S	Bot Charging	0.004	0.000	0.007	0.006	S	Player Pickup (Slow)	0.007	0.010	0.010	0.000
S	Player Proj. Dist.	0.003	0.000	0.006	0.005	S	Bot Proj. Player Dist.	0.007	0.000	0.009	0.011
S	Player Proj. Count	0.003	0.000	0.005	0.005	S	Bot Collision Above	0.005	0.000	0.007	0.008
S	Bot Jumping	0.000	0.000	0.001	0.001	S	Bot Jumping	0.000	0.000	0.000	0.000

A.3.3 Feature Importance in Genre-Based Models for Seen Games

Tables A.13, A.14 and A.15 show μA and ∇A models for racing, shooter, and platformer games respectively. Models are trained on a combined feature set of general and genre-specific features. Rows marked with 'G' indicate general features, and rows marked with 'S' indicate genre-specific features throughout the tables in this section.

Table A.13: MDI feature importance in racing game models trained on 3 games.

μA			∇A		
	Feature	**TinyCars & Solid & ApexSpeed**		**Feature**	**TinyCars & Solid & ApexSpeed**
G	Time Passed	0.136	G	Time Passed	0.050
G	Player Score	0.132	S	Player Crashing	0.049
S	Player Gas Pedal	0.075	G	Player Score	0.046
S	Bot Crashing	0.040	S	Player Speed	0.041
S	Player Lap	0.034	S	Player Speed Boost	0.038
G	Player Activity	0.033	S	Bot Crashing	0.036
S	Bot Score	0.029	G	Player Movement	0.034
S	Player Speed	0.025	G	Object Diversity	0.030
S	Player Standing	0.024	G	Object Intensity	0.030
G	Bot Count	0.023	G	Bot Count	0.028
S	Bot Dist. To WP	0.023	S	Bot Grounded	0.026
S	Bot Standing	0.022	S	Player Standing	0.025
S	Bot Grounded	0.021	S	Player Dist. To WP	0.025
S	Bot Lap	0.021	S	Bot Gas Pedal	0.024
S	Player Crashing	0.019	G	Bot Diversity	0.024
S	Bot Steering	0.019	S	Player Delta Rot.	0.023
G	Bot Movement	0.018	S	Bot Standing	0.022
G	Bot Diversity	0.017	S	Player Looping	0.021
G	Event Intensity	0.017	S	Player Steering	0.021
S	Player Respawn	0.016	S	Bot Player Dist.	0.021
S	Bot Speed	0.016	S	Jump Count	0.020
G	Event Diversity	0.016	G	Event Intensity	0.020
S	Bot Player Dist.	0.016	S	Player Off Road	0.020
S	Player Grounded	0.014	G	Bot Movement	0.020
S	Player Mid Air	0.014	S	Bot Delta Rot.	0.020
S	Bot Gas Pedal	0.014	G	Event Diversity	0.019
S	Bot Delta Rot.	0.014	S	Bot Speed	0.019
G	Player Movement	0.012	G	Player Activity	0.019
S	Player Delta Rot.	0.011	S	Bot Dist. To WP	0.019
S	Player Off Road	0.011	S	Bot Steering	0.019
G	Input Diversity	0.010	S	Player Gas Pedal	0.018
S	Bot Off Road	0.009	G	Input Diversity	0.017
S	Jump Count	0.009	S	Player Mid Air	0.017
S	Player Looping	0.009	S	Player Grounded	0.017

G	Input Intensity	0.009	G	Input Intensity	0.017
S	Player Dist. To WP	0.009	S	Player Lap	0.016
G	Object Intensity	0.009	S	Bot Score	0.015
G	Object Diversity	0.009	S	Obstacle Count	0.015
S	Player Steering	0.007	S	Speed Boost Count	0.013
S	Speed Boost Count	0.007	S	Player Respawn	0.010
S	Player Speed Boost	0.006	S	Loop Count	0.010
S	Bot Speed Boost	0.006	S	Bot Off Road	0.008
S	Bot Looping	0.006	S	Bot Lap	0.008
S	Loop Count	0.005	S	Bot Looping	0.006
S	Obstacle Count	0.004	S	Bot Speed Boost	0.005
S	Bot Respawn	0.002	S	Bot Respawn	0.001

Table A.14: MDI feature importance in shooter game models trained on three game.

μA			∇A		
Feature		**Heist! & TopDown & Shootout**	**Feature**		**Heist! & TopDown & Shootout**
G	Time Passed	0.259	S	Bot Shooting	0.069
G	Player Score	0.183	G	Bot Diversity	0.054
S	Bot Reloading	0.052	S	Bot Proj. Count	0.054
S	Player Health	0.037	S	Bot Proj. Player Dist.	0.052
G	Bot Diversity	0.037	S	Bot Health	0.049
G	Event Diversity	0.028	S	Player Damaged	0.047
G	Event Intensity	0.023	S	Player Aim (Enemy)	0.044
S	Player Damaged	0.023	S	Bot Aim (Player)	0.038
S	Bot Health	0.021	G	Bot Count	0.035
S	Bot Speed X	0.021	G	Bot Movement	0.031
S	Bot Aim (Player)	0.019	G	Event Diversity	0.026
S	Player Pickup (Health)	0.018	S	Bot Speed Z	0.022
S	Player Reloading	0.017	S	Bot Damaged	0.021
S	Player Proj. Count	0.016	S	Player Proj. Count	0.021
S	Bot Shooting	0.015	S	Bot Speed X	0.021
S	Player Healing	0.014	S	Bot Delta Rot.	0.020
S	Player Kill Count	0.013	G	Time Passed	0.020
S	Bot Delta Rot.	0.013	S	Player Proj. Dist.	0.019
S	Player Proj. Dist.	0.012	S	Player Health	0.019
G	Player Activity	0.012	S	Player Reloading	0.019
S	Bot Proj. Count	0.011	S	Player Kill Count	0.019
S	Bot Speed Z	0.011	G	Input Diversity	0.018
S	Bot Speed Y	0.011	G	Event Intensity	0.018
S	Bot Proj. Player Dist.	0.010	G	Input Intensity	0.017
S	Bot Damaged	0.010	G	Player Activity	0.017
S	Player Aim (Enemy)	0.010	G	Player Score	0.017

G	Bot Movement	0.010	S	Player Shooting	0.017
G	Bot Count	0.010	S	Player Delta Rot.	0.016
S	Player Shooting	0.009	S	Reticle Delta Dist.	0.016
S	Player Death	0.007	G	Player Movement	0.016
S	Player Speed Y	0.007	S	Player Speed Z	0.013
G	Input Intensity	0.007	S	Player Speed X	0.013
G	Input Diversity	0.006	S	Player Healing	0.012
S	Player Delta Rot.	0.005	S	Bot Speed Y	0.010
S	Player Speed Z	0.005	S	Player Death	0.010
S	Reticle Delta Dist.	0.004	S	Player Speed Y	0.010
S	Player Speed X	0.004	S	Pick UP Disctance	0.010
S	Pick UP Disctance	0.004	G	Object Intensity	0.008
G	Player Movement	0.004	S	Player Pickup (Health)	0.008
S	Objects Destroyed	0.004	S	Player Aim (Destructible)	0.008
G	Object Diversity	0.003	S	Pickups	0.008
S	Destructible Count	0.003	G	Object Diversity	0.008
G	Object Intensity	0.003	S	Destructible Count	0.007
S	Player Crouching	0.003	S	Bot Reloading	0.007
S	Player Aim (Destructible)	0.003	S	Objects Destroyed	0.006
S	Pickups	0.003	S	Player Crouching	0.005
S	Player Sprinting	0.002	S	Player Sprinting	0.003
S	Player Shoot On Reload	0.001	S	Player Shoot On Reload	0.002

Table A.15: MDI feature importance in platformer game models trained on three game.

μA			∇A		
Feature		**Endless & Pirates! & Run'N'Gun**	**Feature**		**Endless & Pirates! & Run'N'Gun**
G	Time Passed	0.163	G	Bot Movement	0.060
G	Player Score	0.149	G	Bot Count	0.048
S	Player Damaged	0.058	S	Player Health	0.046
S	Player Death	0.039	G	Bot Diversity	0.037
S	Player Coll. Left	0.029	S	Bot Health	0.031
S	Bot Coll. Left	0.029	S	Bot Player Dist.	0.028
G	Player Movement	0.025	S	Player Speed X	0.028
S	Player Health	0.024	S	Player Death	0.027
G	Bot Diversity	0.022	S	Player Pickup (Boost)	0.025
G	Bot Movement	0.022	S	Bot Speed X	0.024
S	Player Coll. Right	0.022	S	Bot Grounded	0.023
S	Player Pickup (Health)	0.021	G	Event Diversity	0.023
S	Bot Speed X	0.019	S	Bot Coll. Below	0.022
S	Bot Speed Y	0.016	G	Player Movement	0.021
S	Player Speed X	0.016	S	Pick UP Disctance	0.019
S	Player Coll. Above	0.015	G	Time Passed	0.019
S	Bot Grounded	0.014	S	Player Damaged	0.019

S	Bot Health	0.013	G	Object Diversity	0.019
S	Bot Player Dist.	0.012	S	Player Speed Y	0.019
S	Bot Coll. Below	0.012	S	Bot Falling	0.018
S	Player Pickup (Point)	0.011	S	Bot Shooting	0.018
S	Bot Proj. Player Dist.	0.011	S	Bot Speed Y	0.018
S	Player Kill Count	0.011	S	Player Falling	0.018
G	Event Diversity	0.011	S	Player Pickup (Point)	0.017
G	Bot Count	0.011	G	Event Intensity	0.017
S	Pick UP Disctance	0.010	G	Input Intensity	0.017
G	Event Intensity	0.010	S	Player Kill Count	0.017
S	Bot Coll. Right	0.010	S	Pickups	0.016
S	Bot Proj. Count	0.009	G	Player Score	0.016
G	Object Intensity	0.009	G	Player Activity	0.016
G	Object Diversity	0.009	G	Object Intensity	0.016
S	Player Speed Y	0.009	G	Input Diversity	0.016
S	Bot Falling	0.009	S	Bot Coll. Left	0.016
S	Pickups	0.009	S	Bot Coll.	0.015
S	Bot Shooting	0.009	S	Player Coll.	0.015
G	Player Activity	0.009	S	Player Jumping	0.015
G	Input Intensity	0.009	S	Player Coll. Above	0.015
S	Player Coll.	0.009	S	Player Coll. Right	0.014
S	Bot Coll.	0.008	S	Player Grounded	0.014
S	Player Pickup (Power)	0.008	S	Player Coll. Below	0.014
S	Bot Coll. Above	0.008	S	Player Shooting	0.013
S	Player Shooting	0.008	S	Player Coll. Left	0.012
G	Input Diversity	0.008	S	Bot Proj. Count	0.010
S	Player Falling	0.008	S	Bot Charging	0.010
S	Player Pickup (Boost)	0.007	S	Bot Damaged	0.010
S	Player Coll. Below	0.007	S	Player Pickup (Health)	0.009
S	Player Grounded	0.007	S	Player Pickup (Slow)	0.008
S	Player Pickup (Slow)	0.006	S	Player Proj. Dist.	0.008
S	Player on Power	0.006	S	Bot Coll. Right	0.008
S	Player Jumping	0.006	S	Player Proj. Count	0.008
S	Bot Charging	0.005	S	Bot Proj. Player Dist.	0.008
S	Bot Damaged	0.005	S	Player Pickup (Power)	0.008
S	Player Proj. Count	0.003	S	Player on Power	0.007
S	Player Proj. Dist.	0.003	S	Bot Coll. Above	0.005
S	Bot Jumping	0.000	S	Bot Jumping	0.000

A.3.4 Feature Importance in General Models for Unseen Genres

Table A.16 show μA and ∇A models for general models trained on 6 games from two genres. Models are trained on general features.

Table A.16: MDI feature importance in general models trained on 6 games from two genres.

μA						∇A					
Feature	**Racing & Shooter**	**Feature**	**Racing & Platfomer**	**Feature**	**Shooter & Platfomer**	**Feature**	**Racing & Shooter**	**Feature**	**Racing & Platfomer**	**Feature**	**Shooter & Platfomer**
T Passed	0.378	T Passed	0.322	T Passed	0.382	B Diversity	0.137	B Count	0.124	B Diversity	0.159
P Score	0.261	P Score	0.269	P Score	0.272	B Count	0.102	B Diversity	0.106	B Move.	0.148
B Diversity	0.051	P Move.	0.055	B Diversity	0.064	P Score	0.088	B Move.	0.104	B Count	0.118
P Activity	0.050	B Move.	0.049	E Diversity	0.047	T Passed	0.086	P Move.	0.090	E Diversity	0.093
E Diversity	0.044	P Activity	0.046	E Intensity	0.041	E Diversity	0.085	O Diversity	0.084	E Intensity	0.066
E Intensity	0.042	B Diversity	0.044	B Move.	0.039	B Move.	0.081	O Intensity	0.083	T Passed	0.063
B Count	0.037	E Intensity	0.039	B Count	0.027	P Move.	0.069	T Passed	0.071	P Move.	0.056
B Move.	0.033	B Count	0.037	P Move.	0.025	O Intensity	0.067	P Score	0.063	P Score	0.055
I Diversity	0.023	E Diversity	0.035	P Activity	0.024	E Intensity	0.064	E Diversity	0.061	I Diversity	0.051
I Intensity	0.021	O Diversity	0.028	O Intensity	0.022	O Diversity	0.062	P Activity	0.059	O Diversity	0.049
O Intensity	0.021	O Intensity	0.028	O Diversity	0.020	P Activity	0.057	E Intensity	0.055	P Activity	0.049
P Move.	0.020	I Diversity	0.025	I Intensity	0.019	I Diversity	0.056	I Diversity	0.051	I Intensity	0.047
O Diversity	0.019	I Intensity	0.024	I Diversity	0.018	I Intensity	0.048	I Intensity	0.049	O Intensity	0.046

A.3.5 Feature Importance in General Models for Unseen Games

Tables A.17 and A.18 show μA and ∇A models, respectively, for general models trained on 8 games. Models are trained on general features.

Table A.17: MDI feature importance in general μA models trained on 8 games.

	μA									
Feature	**General Average**	**All Except TinyCars**	**All Except Solid**	**All Except ApexSpeed**	**All Except Heist!**	**All Except TopDown**	**All Except Shootout**	**All Except Endless**	**All Except Pirates!**	**All Except Run'N'Gun**
Time Passed	0.377	0.402	0.390	0.381	0.364	0.367	0.367	0.380	0.374	0.367
Player Score	0.286	0.286	0.278	0.277	0.295	0.285	0.290	0.284	0.290	0.289
Bot Diversity	0.046	0.040	0.046	0.053	0.049	0.046	0.040	0.049	0.045	0.044
Event Diversity	0.041	0.049	0.045	0.038	0.036	0.041	0.041	0.039	0.042	0.040
Event Intensity	0.039	0.041	0.038	0.035	0.037	0.037	0.044	0.040	0.041	0.042
Player Activity	0.038	0.029	0.027	0.042	0.042	0.032	0.043	0.044	0.044	0.041
Bot Movement	0.037	0.038	0.038	0.035	0.038	0.043	0.033	0.039	0.032	0.035
Bot Count	0.031	0.025	0.032	0.032	0.031	0.034	0.032	0.033	0.030	0.029
Player Movement	0.026	0.022	0.025	0.025	0.028	0.033	0.028	0.018	0.026	0.029
Object Intensity	0.022	0.018	0.024	0.023	0.022	0.022	0.023	0.021	0.020	0.024
Object Diversity	0.021	0.017	0.022	0.022	0.021	0.023	0.022	0.019	0.020	0.022
Input Diversity	0.019	0.017	0.017	0.019	0.018	0.019	0.020	0.018	0.019	0.020
Input Intensity	0.018	0.018	0.017	0.018	0.018	0.018	0.019	0.017	0.018	0.019

Table A.18: MDI feature importance in general ∇A models trained on 8 games.

	∇A									
Feature	**General Average**	**All Except TinyCars**	**All Except Solid**	**All Except ApexSpeed**	**All Except Heist!**	**All Except TopDown**	**All Except Shootout**	**All Except Endless**	**All Except Pirates!**	**All Except Run'N'Gun**
Bot Diversity	0.140	0.147	0.147	0.145	0.118	0.131	0.145	0.140	0.146	0.142
Bot Count	0.124	0.119	0.126	0.129	0.116	0.130	0.134	0.126	0.121	0.116
Bot Movement	0.110	0.122	0.117	0.120	0.103	0.119	0.104	0.080	0.103	0.118
Event Diversity	0.082	0.089	0.086	0.083	0.083	0.065	0.083	0.091	0.083	0.077
Player Movement	0.069	0.068	0.060	0.063	0.078	0.074	0.068	0.065	0.073	0.069
Time Passed	0.069	0.065	0.068	0.065	0.070	0.069	0.065	0.083	0.067	0.064
Object Intensity	0.067	0.063	0.059	0.065	0.076	0.074	0.069	0.061	0.069	0.072
Object Diversity	0.067	0.064	0.059	0.064	0.078	0.073	0.068	0.056	0.070	0.072
Player Score	0.065	0.060	0.062	0.061	0.063	0.064	0.063	0.079	0.067	0.064
Event Intensity	0.060	0.060	0.063	0.062	0.062	0.052	0.058	0.065	0.059	0.054
Player Activity	0.053	0.050	0.055	0.050	0.055	0.055	0.052	0.053	0.051	0.055
Input Diversity	0.050	0.048	0.051	0.048	0.050	0.050	0.048	0.053	0.047	0.052
Input Intensity	0.045	0.045	0.047	0.044	0.047	0.046	0.043	0.047	0.043	0.045

Bibliography

Mojtaba Khomami Abadi, Ramanathan Subramanian, Seyed Mostafa Kia, Paolo Avesani, Ioannis Patras, and Nicu Sebe. Decaf: Meg-based multimodal database for decoding affective physiological responses. *IEEE Transactions on Affective Computing*, 6(3):209–222, 2015. — Cited on page 21.

Amir Zaib Abbasi, Ding Hooi Ting, Helmut Hlavacs, Liliana Vale Costa, and Ana Isabel Veloso. An empirical validation of consumer video game engagement: A playful-consumption experience approach. *Entertainment Computing*, 29:43–55, 2019. — Cited on pages 13 and 45.

Shazia Afzal and Peter Robinson. Natural affect data: Collection and annotation. In *New perspectives on affect and learning technologies*, pages 55–70. Springer, 2011. — Cited on page 22.

Renan Vinicius Aranha, Cléber Gimenez Corrêa, and Fátima LS Nunes. Adapting software with affective computing: a systematic review. *IEEE Transactions on Affective Computing*, 2019. — Cited on page 11.

Stylianos Asteriadis, Noor Shaker, Kostas Karpouzis, and Georgios N Yannakakis. Towards player's affective and behavioral visual cues as drives to game adaptation. In *Multimodal Corpora: How Should Multimodal Corpora Deal with the Situation? Workshop Programme*. LREC, 2012. — Cited on pages 12, 14, and 16.

Hillel Aviezer, Ran R Hassin, Jennifer Ryan, Cheryl Grady, Josh Susskind, Adam Anderson, Morris Moscovitch, and Shlomo Bentin. Angry, disgusted, or afraid? studies on the malleability of emotion perception. *Psychological science*, 19(7):724–732, 2008. — Cited on page 12.

Hillel Aviezer, Yaacov Trope, and Alexander Todorov. Body cues, not facial expressions, discriminate between intense positive and negative emotions. *Science*, 338(6111):1225–1229, 2012. — Cited on page 12.

Ahmad Azadvar and Alessandro Canossa. Upeq: ubisoft perceived experience questionnaire: a self-determination evaluation tool for video games. In *Proceedings of the International Conference on the Foundations of Digital Games (FDG)*. ACM, 2018. — Cited on page 16.

Sander CJ Bakkes, Pieter HM Spronck, and Giel van Lankveld. Player behavioural modelling for video games. *Entertainment Computing*, 3(3):71–79, 2012. — Cited on pages 14 and 45.

Fabrizio Balducci, Costantino Grana, and Rita Cucchiara. Affective level design for a role-playing videogame evaluated by a brain–computer interface and machine learning methods. *The Visual Computer*, 33(4):413–427, 2017. — Cited on page 12.

Lisa Feldman Barrett, Batja Mesquita, and Maria Gendron. Context in emotion perception. *Current Directions in Psychological Science*, 20(5):286–290, 2011. — Cited on page 12.

Lisa Feldman Barrett, Ralph Adolphs, Stacy Marsella, Aleix M Martinez, and Seth D Pollak. Emotional expressions reconsidered: Challenges to inferring emotion from human facial movements. *Psychological science in the public interest*, 20(1):1–68, 2019. — Cited on page 12.

Christian Bauckhage, Anders Drachen, and Rafet Sifa. Clustering game behavior data. *IEEE Transactions on Computational Intelligence and AI in Games*, 7(3):266–278, 2015. — Cited on page 13.

Yoann Baveye, Emmanuel Dellandréa, Christel Chamaret, and Liming Chen. Deep learning vs. kernel methods: Performance for emotion prediction in videos. In *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 77–83. IEEE, 2015a. — Cited on pages 19 and 37.

Yoann Baveye, Emmanuel Dellandrea, Christel Chamaret, and Liming Chen. Liris-accede: A video database for affective content analysis. *IEEE Transactions on Affective Computing*, 6(1):43–55, 2015b. — Cited on page 21.

Nicolas Beaudoin-Gagnon, Alexis Fortin-Côté, Cindy Chamberland, Ludovic Lefebvre, Jérémy Bergeron-Boucher, Alexandre Campeau-Lecours, Sébastien Tremblay, and Philip L Jackson. The funii database: A physiological, behavioral, demographic and subjective video game database for affective gaming and player experience research. In *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 1–7. IEEE, 2019. — Cited on pages 21 and 22.

Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013. — Cited on page 14.

Donald J Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA, USA:, 1994. — Cited on pages 53 and 54.

Valerio Bonometti, Charles Ringer, Mathieu Ruiz, Alex Wade, and Anders Drachen. From theory to behaviour: Towards a general model of engagement. *arXiv preprint arXiv:2004.12644*, 2020. — Cited on pages 11, 14, 15, and 52.

Brandon M Booth. *Improving Modeling of Human Experience and Behavior: Methodologies for Enhancing the Quality of Human-Produced Data and Annotations of Subjective Constructs.* PhD thesis, University of Southern California, 2020. — Cited on page 101.

Margaret M Bradley and Peter J Lang. Measuring emotion: the self-assessment manikin and the semantic differential. *Journal of behavior therapy and experimental psychiatry*, 25(1):49–59, 1994. — Cited on page 16.

Leo Breiman. *Classification and regression trees*. Routledge, 2017. — Cited on page 29.

Joost Broekens and Willem-Paul Brinkman. Affectbutton: A method for reliable and valid affective self-report. *International Journal of Human-Computer Studies*, 71(6):641–667, 2013. — Cited on pages 17, 18, 19, and 31.

David Buckingham and Andrew Burn. Game literacy in theory and practice. *Journal of Educational Multimedia and Hypermedia*, 16(3):323–349, 2007. — Cited on page 46.

Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*, pages 89–96, 2005. — Cited on pages 24 and 27.

Christopher JC Burges. From ranknet to lambdarank to lambdamart: An overview. *Learning*, 11(23-581):81, 2010. — Cited on page 24.

Erik Cambria, Andrew Livingstone, and Amir Hussain. The hourglass of emotions. In *Cognitive behavioural systems*, pages 144–157. Springer, 2012. — Cited on page 12.

Elizabeth Camilleri, Georgios N Yannakakis, and Antonios Liapis. Towards general models of player affect. In *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 333–339, 2017. — Cited on pages 2, 14, 15, 17, 20, 22, 24, 29, 40, 41, 43, 45, 52, 57, 59, 63, and 73.

Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*, pages 129–136, 2007. — Cited on page 25.

Guillaume Chanel, Cyril Rebetez, Mireille Bétrancourt, and Thierry Pun. Boredom, engagement and anxiety as indicators for adaptation to difficulty in games. In *Proceedings of the international conference on Entertainment and media in the ubiquitous era*, pages 13–17, 2008. — Cited on page 12.

Guillaume Chanel, Cyril Rebetez, Mireille Bétrancourt, and Thierry Pun. Emotion assessment from physiological signals for adaptation of game difficulty. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 41(6):1052–1063, 2011. — Cited on page 12.

Despoina Chatzakou, Athena Vakali, and Konstantinos Kafetsios. Detecting variation of emotions in online activities. *Expert Systems with Applications*, 89:318–332, 2017. — Cited on page 29.

Andrea Clerico, Cindy Chamberland, Mark Parent, Pierre-Emmanuel Michon, Sebastien Tremblay, Tiago H Falk, Jean-Christophe Gagnon, and Philip Jackson. Biometrics and classifier fusion to predict the fun-factor in video gaming. In *2016 IEEE Conference on Computational Intelligence and Games (CIG)*, pages 1–8. IEEE, 2016. — Cited on pages 13 and 45.

Corinna Cortes and Vladimir Vapnik. Support-vector network. *Machine Learning*, 20: 273–297, 1995. — Cited on page 27.

Alan S Cowen and Dacher Keltner. Self-report captures 27 distinct categories of emotion bridged by continuous gradients. *Proceedings of the National Academy of Sciences*, 114 (38):E7900–E7909, 2017. — Cited on page 12.

Roddy Cowie and Randolph R Cornelius. Describing the emotional states that are expressed in speech. *Speech communication*, 40(1-2):5–32, 2003. — Cited on page 12.

Roddy Cowie, Ellen Douglas-Cowie, Susie Savvidou*, Edelle McMahon, Martin Sawey, and Marc Schröder. 'FEELTRACE': An instrument for recording perceived emotion in real time. In *ISCA tutorial and research workshop (ITRW) on speech and emotion*, 2000. — Cited on pages 13, 17, 18, 19, and 31.

Roddy Cowie, Martin Sawey, Cian Doherty, Javier Jaimovich, Cavan Fyans, and Paul Stapleton. Gtrace: General trace program compatible with EmotionML. In *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 709–710. IEEE, 2013. — Cited on pages 13, 18, 19, and 31.

Antonio R Damasio. *Descartes' error: Emotion, rationality and the human brain.* New York: Putnam, 1994. — Cited on pages 1, 16, 17, 19, and 52.

Emmanuel Dellandréa, Liming Chen, Yoann Baveye, Mats Viktor Sjöberg, Christel Chamaret, et al. The mediaeval 2016 emotional impact of movies task. In *CEUR Workshop Proceedings*, 2016. — Cited on page 19.

Svati Dhamija and Terrance E Boult. Automated action units vs. expert raters: Face off. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 259–268. IEEE, 2018. — Cited on page 19.

Janine Diehl-Schmid, Corina Pohl, Carolin Ruprecht, Stefan Wagenpfeil, Hans Foerstl, and Alexander Kurz. The Ekman 60 faces test as a diagnostic instrument in frontotemporal dementia. *Archives of Clinical Neuropsychology*, 22(4):459–464, 2007. — Cited on page 12.

Anders Drachen and Shawn Connor. Games analytics for games user research. In Anders Drachen, Pejman Mirza-Babaei, and Lennart E Nacke, editors, *Games User Research*, pages 333–355. Oxford University Press, 2018. — Cited on page 16.

Anders Drachen, Alessandro Canossa, and Georgios N Yannakakis. Player modeling using self-organization in Tomb Raider: Underworld. In *Proceedings of the Symposium on Computational Intelligence and Games (CIG)*, pages 1–8. IEEE, 2009. — Cited on page 13.

Anders Drachen, Rafet Sifa, Christian Bauckhage, and Christian Thurau. Guns, swords and data: Clustering of player behavior in computer games in the wild. In *Proceedings of the Conference on Computational Intelligence and Games (CIG)*, pages 163–170, 2012. — Cited on page 13.

Paul Ekman. An argument for basic emotions. *Cognition & emotion*, 6(3-4):169–200, 1992. — Cited on page 12.

Rana El Kaliouby, Rosalind Picard, and Simon Baron-Cohen. Affective computing and autism. *Annals of the New York Academy of Sciences*, 1093(1):228–248, 2006. — Cited on page 106.

Magy Seif El-Nasr, Anders Drachen, and Alessandro Canossa. *Game analytics*. Springer, 2016. — Cited on pages 13 and 16.

Susanne Erk, Markus Kiefer, Jo Grothe, Arthur P Wunderlich, Manfred Spitzer, and Henrik Walter. Emotional context modulates subsequent memory effect. *Neuroimage*, 18(2):439–447, 2003. — Cited on pages 17, 19, and 52.

Vincent E Farrugia, Héctor P Martínez, and Georgios N Yannakakis. The preference learning toolbox. *arXiv preprint arXiv:1506.01709*, 2015. — Cited on pages 25, 26, and 27.

Basura Fernando, Amaury Habrard, Marc Sebban, and Tinne Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *Proceedings of the IEEE international conference on computer vision*, pages 2960–2967, 2013. — Cited on page 15.

Alexis Fortin-Côté, Cindy Chamberland, Mark Parent, Sébastien Tremblay, Philip Jackson, Nicolas Beaudoin-Gagnon, Alexandre Campeau-Lecours, Jérémy Bergeron-Boucher, and Ludovic Lefebvre. Predicting video game players' fun from physiological and behavioural data. In *Future of Information and Communication Conference*, pages 479–495. Springer, 2018. — Cited on pages 12 and 16.

Julian Frommel, Claudia Schrader, and Michael Weber. Towards emotion-based adaptive games: Emotion recognition via input and performance features. In *Proceedings of the Annual Symposium on Computer-Human Interaction in Play (CHI Play)*, pages 173–185. ACM, 2018. — Cited on page 16.

Johannes Fürnkranz and Eyke Hüllermeier. Pairwise preference learning and ranking. In *Proceedings of the European Conference on Machine Learning*, pages 145–156. Springer, 2003. — Cited on page 24.

Johannes Fürnkranz and Eyke Hüllermeier. Preference learning. In *Encyclopedia of Machine Learning*, pages 789–795. Springer, 2011. — Cited on pages 23 and 28.

Daniel Gabana, Laurissa Tokarchuk, Emily Hannon, and Hatice Gunes. Effects of valence and arousal on working memory performance in virtual reality gaming. In *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 36–41. IEEE, 2017. — Cited on page 45.

Vanessa Georges, François Courtemanche, Marc Fredette, Pierre-Majorique Léger, and Sylvain Sénécal. Developing personas based on physiological measures. In *Proceedings of the International Conference on Physiological Computing Sysytems*, 2018. — Cited on page 17.

Jeffrey M Girard. Carma: Software for continuous affect rating and media annotation. *Journal of Open Research Software*, 2(1), 2014. — Cited on pages 18, 19, and 31.

Jeffrey M Girard and Aidan GC Wright. Darma: Software for dual axis rating and media annotation. *Behavior research methods*, 50(3):902–909, 2018. — Cited on pages 18 and 19.

Amogh Gulati, Brihi Joshi, Chirag Jain, and Jainendra Shukla. It's not what they play, it's what you hear: Understanding perceived vs. induced emotions in hindustani classical music. In *Companion Publication of the 2020 International Conference on Multimodal Interaction*, pages 42–46, 2020. — Cited on page 101.

Hatice Gunes and Björn Schuller. Categorical and dimensional affect analysis in continuous input: Current trends and future directions. *Image and Vision Computing*, 31(2):120–136, 2013. — Cited on page 19.

Sharath Chandra Guntuku, Michael James Scott, Huan Yang, Gheorghita Ghinea, and Weisi Lin. The cp-qae-i: A video dataset for exploring the effect of personality and culture on perceived quality and affect in multimedia. In *Proceedings of the International Workshop on Quality of Multimedia Experience (QoMEX)*, pages 1–7. IEEE, 2015. — Cited on page 21.

Salma Hamdy and David King. Affect and believability in game characters–a review of the use of affective computing in games. In *Proceedings of the 18th Annual Conference on Simulation and AI in Computer Games. EUROSIS*, 2017. — Cited on page 11.

Jing Han, Xue Li, Lun Xie, Jing Liu, Feifei Wang, and Zhiliang Wang. Affective computing of childern with authism based on feature transfer. In *Proceedings of the IEEE International Conference on Cloud Computing and Intelligence Systems (CCIS)*, pages 845–849. IEEE, 2018. — Cited on page 106.

Dilana Hazer-Rau, Lin Zhang, and Harald C Traue. A workflow for affective computing and stress recognition from biosignals. In *Proceedings of the Electronic Conference on Sensors and Applications*, volume 15, page 30, 2020. — Cited on page 29.

Alexander Heimerl, Tobias Baur, Florian Lingenfelser, Johannes Wagner, and Elisabeth André. Nova-a tool for explainable cooperative machine learning. In *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 109–115. IEEE, 2019. — Cited on pages 18, 20, and 104.

Harry Helson. *Adaptation-level theory: an experimental and systematic approach to behavior.* Harper and Row: New York, 1964. — Cited on pages 17 and 19.

Geoffrey E. Hinton and R. S. Zemel. Autoencoders, minimum description length, and helmholtz free energy. In *Advances in Neural Information Processing Systems*, pages 3–10, 1994. — Cited on page 25.

Christoffer Holmgård, Georgios N Yannakakis, Karen-Inge Karstoft, and Henrik Steen Andersen. Stress detection for ptsd via the startlemart game. In *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 523–528. IEEE, 2013. — Cited on page 106.

Christoffer Holmgård, Antonios Liapis, Julian Togelius, and Georgios N Yannakakis. Evolving personas for player decision modeling. In *Proceedings of the Conference on Computational Intelligence and Games (CIG)*, 2014. — Cited on page 13.

WA IJsselsteijn, YAW De Kort, and Karolien Poels. The game experience questionnaire. *Eindhoven: Technische Universiteit Eindhoven*, 2013. — Cited on page 16.

Mimansa Jaiswal, Zakaria Aldeneh, Cristian-Paul Bara, Yuanhang Luo, Mihai Burzo, Rada Mihalcea, and Emily Mower Provost. Muse-ing on the impact of utterance ordering on crowdsourced emotion annotations. *arXiv preprint arXiv:1903.11672*, 2019. — Cited on page 41.

Thorsten Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 133–142. ACM, 2002. — Cited on page 23.

Christian Jones and Jamie Sutherland. Acoustic emotion recognition for affective computer gaming. In *Affect and emotion in human-computer interaction*, pages 209–219. Springer, 2008. — Cited on page 12.

Markus Kächele, Martin Schels, and Friedhelm Schwenker. The influence of annotation, corpus design, and evaluation on the outcome of automatic classification of human emotions. *Frontiers in Information and Communication Technology*, 3:27, 2016. — Cited on page 33.

Haik Kalantarian, Khaled Jedoui, Peter Washington, and Dennis P Wall. A mobile game for automatic emotion-labeling of images. *IEEE transactions on games*, 12(2):213–218, 2018. — Cited on page 12.

Kostas Karpouzis, Georgios N Yannakakis, Noor Shaker, and Stylianos Asteriadis. The platformer experience dataset. In *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 712–718. IEEE, 2015. — Cited on pages 21 and 22.

Michael Kipp. Anvil-a generic annotation tool for multimodal dialogue. In *Seventh European Conference on Speech Communication and Technology*, 2001. — Cited on pages 17 and 18.

Madison Klarkowski, Daniel Johnson, Peta Wyeth, Cody Phillips, and Simon Smith. Psychophysiology of challenge in play: Eda and self-reported arousal. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, pages 1930–1936, 2016. — Cited on pages 13 and 45.

Sander Koelstra, Christian Muhl, Mohammad Soleymani, Jong-Seok Lee, Ashkan Yazdani, Touradj Ebrahimi, Thierry Pun, Anton Nijholt, and Ioannis Patras. DEAP: A database for emotion analysis using physiological signals. *IEEE Transactions on Affective Computing*, 3(1):18–31, 2012. — Cited on pages 21 and 22.

Agata Kołakowska, Agnieszka Landowska, Mariusz Szwoch, Wioleta Szwoch, and Michał R Wróbel. Emotion recognition and its application in software engineering. In *Proceedings of the International Conference on Human System Interactions*, pages 532–539. IEEE, 2013. — Cited on page 16.

Jean Kossaifi, Robert Walecki, Yannis Panagakis, Jie Shen, Maximilian Schmitt, Fabien Ringeval, Jing Han, Vedhas Pandit, Antoine Toisoul, Bjoern W Schuller, et al. Sewa db: A rich database for audio-visual emotion and sentiment research in the wild. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019. — Cited on pages 21 and 22.

Bernhard Kratzwald, Suzana Ilić, Mathias Kraus, Stefan Feuerriegel, and Helmut Prendinger. Deep learning for affective computing: Text-based emotion recognition in decision support. *Decision Support Systems*, 115:24–35, 2018. — Cited on page 29.

Klaus Krippendorff. Reliability in content analysis. *Human communication research*, 30(3): 411–433, 2004. — Cited on page 40.

Klaus Krippendorff. Computing krippendorff's alpha-reliability. `https://repository.upenn.edu/asc_papers/43`, 2011. Accessed 15 April 2019. — Cited on page 40.

Petri Lankoski and Staffan Björk. *Game Research Methods. An Overview.* ETC Press, 2015. — Cited on page 33.

Richard S Lazarus and Bernice N Lazarus. *Passion and reason: Making sense of our emotions.* Oxford University Press, USA, 1996. — Cited on page 12.

Roger J Lewis. An introduction to classification and regression tree (cart) analysis. In *Proceedings of the society for Academic Emergency Medicine (SAEM) annual meeting*, 2000. — Cited on page 29.

Antonios Liapis, Georgios N Yannakakis, Mark J Nelson, Mike Preuss, and Rafael Bidarra. Orchestrating game generation. *IEEE Transactions on Games*, 11(1):48–68, 2018. — Cited on page 45.

Jeroen Lichtenauer and Mohammad Soleymani. Mahnob-hci-tagging database, 2011. — Cited on pages 21 and 22.

Rensis Likert. A technique for the measurement of attitudes. *Archives of psychology*, 1932. — Cited on page 19.

Chong-U Lim, Antonios Liapis, and D. Fox Harrell. Discovering social and aesthetic categories of avatars: A bottom-up artificial intelligence approach using image clustering. In *Proceedings of the International Joint Conference of DiGRA and FDG*, 2016. — Cited on page 13.

Tie-Yan Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 2011. — Cited on pages 23 and 24.

Xi Liu, Muhe Xie, Xidao Wen, Rui Chen, Yong Ge, Nick Duffield, and Na Wang. A semi-supervised and inductive embedding model for churn prediction of large-scale mobile games. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pages 277–286. IEEE, 2018. — Cited on page 14.

Wei-Yin Loh. Classification and regression trees. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(1):14–23, 2011. — Cited on page 29.

Phil Lopes, Antonios Liapis, and Georgios N Yannakakis. Modelling affect for horror soundscapes. *IEEE Transactions on Affective Computing*, 10(2):209–222, 2017a. — Cited on pages 13, 21, 24, 45, 46, and 63.

Phil Lopes, Georgios N Yannakakis, and Antonios Liapis. Ranktrace: Relative and unbounded affect annotation. In *Proceedings of the International Conference on Affective Computing and Intelligent Interaction*, pages 158–163. IEEE, 2017b. — Cited on pages 13, 18, 19, 20, 23, 31, 32, 35, 37, 38, 40, 44, 45, 52, and 59.

Reza Lotfian and Carlos Busso. Practical considerations on the use of preference learning for ranking emotional speech. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 5205–5209. IEEE, 2016. — Cited on page 17.

Gilles Louppe, Louis Wehenkel, Antonio Sutera, and Pierre Geurts. Understanding variable importances in forests of randomized trees. *Advances in neural information processing systems 26*, 2013. — Cited on page 29.

Tobias Mahlmann, Anders Drachen, Julian Togelius, Alessandro Canossa, and Georgios N Yannakakis. Predicting player behavior in Tomb Raider: Underworld. In *Proceedings of the Symposium on Computational Intelligence and Games (CIG)*, pages 178–185. IEEE, 2010. — Cited on page 14.

Konstantinos Makantasis, Antonios Liapis, and Georgios N Yannakakis. From pixels to affect: a study on games and player experience. In *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 1–7. IEEE, 2019. — Cited on pages 14, 16, 43, 45, and 51.

Konstantinos Makantasis, Antonios Liapis, and Georgios N Yannakakis. The pixels and sounds of emotion: General-purpose representations of arousal in games. *IEEE Transactions on Affective Computing*, 2021. — Cited on pages 14, 16, 101, and 105.

Sasha Makarovych, Alessandro Canossa, Julian Togelius, and Anders Drachen. Like a dna string: Sequence-based player profiling in Tom Clancy's The Division. In *Proceedings of the Artificial Intelligence and Interactive Digital Entertainment Conference*. York, 2018. — Cited on page 14.

Soroosh Mariooryad and Carlos Busso. Analysis and compensation of the reaction lag of evaluators in continuous emotional annotations. In *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 85–90. IEEE, 2013. — Cited on page 59.

Héctor P Martínez and Georgios N Yannakakis. Mining multimodal sequential patterns: a case study on affect detection. In *Proceedings of the International Conference on Multimodal Interfaces*, pages 3–10. ACM, 2011. — Cited on page 13.

Héctor P Martínez and Georgios N Yannakakis. Deep multimodal fusion: Combining discrete events and continuous signals. In *Proceedings of the International Conference on Multimodal Interaction*, pages 34–41. ACM, 2014. — Cited on pages 51 and 73.

Hector P Martinez, Yoshua Bengio, and Georgios N Yannakakis. Learning deep physiological models of affect. *IEEE Computational intelligence magazine*, 8(2):20–33, 2013. — Cited on page 16.

Hector P Martinez, Georgios N Yannakakis, and John Hallam. Don't classify ratings of affect; rank them! *IEEE Transactions on Affective Computing*, 5(3):314–326, 2014. — Cited on pages 13, 17, 51, and 52.

Héctor Perez Martínez, Maurizio Garbarino, and Georgios N Yannakakis. Generic physiological features as predictors of player experience. In *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 267–276. Springer, 2011. — Cited on pages 11, 16, 17, and 73.

Daniel McDuff, Rana Kaliouby, Thibaud Senechal, May Amr, Jeffrey Cohn, and Rosalind Picard. Affectiva-MIT facial expression dataset (AM-FED): Naturalistic and spontaneous facial expressions collected "in-the-wild". In *Proceedings of the IEEE Conference on*

Computer Vision and Pattern Recognition Workshops, pages 881–888, 2013. — Cited on page 31.

Gary McKeown, Michel Valstar, Roddy Cowie, Maja Pantic, and Marc Schroder. The SEMAINE database: Annotated multimodal records of emotionally colored conversations between a person and a limited agent. *IEEE Transactions on Affective Computing*, 3(1): 5–17, 2012. — Cited on page 39.

Albert Mehrabian. *Basic dimensions for a general psychological theory implications for personality, social, environmental, and developmental studies.* Cambridge, 1980. — Cited on pages 12 and 13.

Elisa D Mekler, Julia Ayumi Bopp, Alexandre N Tuch, and Klaus Opwis. A systematic review of quantitative studies on the enjoyment of digital entertainment games. In *Proceedings of the Human Factors in Computing Systems conference*, pages 927–936. ACM, 2014. — Cited on page 16.

David Melhart. Towards a comprehensive model of mediating frustration in videogames. *Game Studies*, 18(1), 2018. — Cited on pages 13 and 45.

David Melhart, Ahmad Azadvar, Alessandro Canossa, Antonios Liapis, and Georgios N Yannakakis. Your gameplay says it all: Modelling motivation in Tom Clancy's The Division. In *Proceedings of the IEEE Conference on Games (CoG)*, 2019a. — Cited on pages 17 and 28.

David Melhart, Antonios Liapis, and Georgios N Yannakakis. Pagan: Video affect annotation made easy. In *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, 2019b. — Cited on pages 19, 20, 32, 35, 38, 45, 49, 52, and 101.

David Melhart, Daniele Gravina, and Georgios N Yannakakis. Moment-to-moment engagement prediction through the eyes of the observer: Pubg streaming on twitch. In *Proceedings of the International Conference on the Foundations of Digital Games (FDG)*, pages 1–10, 2020a. — Cited on pages 14, 16, and 45.

David Melhart, Konstantinos Sfikas, Giorgos Giannakakis, and Georgios Yannakakis Antonios Liapis. A study on affect model validity: Nominal vs ordinal labels. In *Workshop on Artificial Intelligence in Affective Computing*, pages 27–34. Proceedings of Machine Learning Research, 2020b. — Cited on pages 17, 23, 24, and 63.

David Melhart, Georgios N Yannakakis, and Antonios Liapis. I feel I feel you: A theory of mind experiment in games. *KI-Künstliche Intelligenz*, 34(1):45–55, 2020c. — Cited on pages 16, 20, 28, and 45.

David Melhart, Antonios Liapis, and Georgios N. Yannakakis. The Affect Game AnnotatIoN (AGAIN) dataset. *arXiv preprint arXiv:2104.02643*, 2021. — Cited on pages 47, 51, 53, and 101.

Vitalik Melnikov, Pritha Gupta, Bernd Frick, Daniel Kaimann, and Eyke Hüllermeier. Pairwise versus pointwise ranking: A case study. *Schedae Informaticae*, 25:73–83, 2016. — Cited on page 24.

Angeliki Metallinou and Shrikanth Narayanan. Annotation and processing of continuous emotional attributes: Challenges and opportunities. In *Proceedings of the IEEE international conference and workshops on automatic face and gesture recognition (FG)*, pages 1–8. IEEE, 2013. — Cited on pages 19 and 59.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015. — Cited on page 43.

Ali Mollahosseini, Behzad Hasani, and Mohammad H Mahoor. Affectnet: A database for facial expression, valence, and arousal computing in the wild. *IEEE Transactions on Affective Computing*, 10(1):18–31, 2017. — Cited on page 21.

Jon D Morris. SAM: the self-assessment manikin. an efficient cross-cultural measurement of emotional response. *Journal of advertising research*, 35(6):63–69, 1995. — Cited on pages 13 and 19.

Philipp M Müller, Sikandar Amin, Prateek Verma, Mykhaylo Andriluka, and Andreas Bulling. Emotion recognition from embedded bodily expressions and speech during dyadic interactions. In *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 663–669. IEEE, 2015. — Cited on page 19.

Frederik Nagel, Reinhard Kopiez, Oliver Grewe, and Eckart Altenmüller. Emujoy: Software for continuous measurement of perceived emotions in music. *Behavior Research Methods*, 39(2):283–290, 2007. — Cited on pages 18 and 19.

Roger B. Nelsen. Kendall tau metric. *Encyclopaedia of mathematics*, 3:226–227, 2001. — Cited on page 60.

Baruch Nevo. Face validity revisited. *Journal of Educational Measurement*, 22(4):287–293, 1985. — Cited on page 13.

Seth Ovadia. Ratings and rankings: Reconsidering the structure of values and their measurement. *International Journal of Social Research Methodology*, 7(5):403–414, 2004. — Cited on page 19.

Cristiana Pacheco, David Melhart, Antonios Liapis, Georgions N. Yannakakis, and Diego Pérez-Liébana. "Trace It Like You Believe It": Towards time-continuous believability prediction. In *Proceedings of the IEEE Conference on Games (CoG)*, in review. — Cited on pages 33 and 101.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. — Cited on page 29.

Diego Perez-Liebana, Spyridon Samothrakis, Julian Togelius, Tom Schaul, and Simon Lucas. General video game ai: Competition, challenges and opportunities. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016a. — Cited on pages 1, 14, and 73.

Diego Perez-Liebana, Spyridon Samothrakis, Julian Togelius Togelius, Tom Schaul, and Simon Lucas. Towards automatic personalized content generation for platform games. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 30, 2016b. — Cited on pages 15 and 43.

África Periáñez, Alain Saas, Anna Guitart, and Colin Magne. Churn prediction in mobile social games: towards a complete assessment using survival ensembles. In *Proceedings of the International Conference on Data Science and Advanced Analytics (DSAA)*, pages 564–573, 2016. — Cited on pages 14 and 16.

Cody Phillips, Daniel Johnson, Madison Klarkowski, Melanie Jade White, and Leanne Hides. The impact of rewards and trait reward responsiveness on player motivation. In *Proceedings of the 2018 Symposium on Computer-Human Interaction in Play*, pages 393–404. ACM, 2018. — Cited on page 16.

Raphael Patrick Prager, Laura Troost, Simeon Brüggenjürgen, David Melhart, Georgios Yannakakis, and Mike Preuss. An experiment on game facet combination. In *Proceedings of the IEEE Conference on Games (CoG)*, 2019. — Cited on page 17.

Jesse J Prinz. *Gut reactions: A perceptual theory of emotion.* Oxford University Press, 2004. — Cited on page 1.

Andrea Amelio Ravelli, Antonio Origlia, and Felice Dell'Orletta. Exploring attention in a multimodal corpus of guided tours. In *Proceedings of the Seventh Italian Conference on Computational Linguistics (CLiC-it 2020), CEUR Workshop Series*, pages 1–6, 2020. — Cited on page 101.

Fabien Ringeval, Andreas Sonderegger, Juergen Sauer, and Denis Lalanne. Introducing the recola multimodal corpus of remote collaborative and affective interactions. In *Proceedings of the IEEE international conference and workshops on automatic face and gesture recognition (FG)*, pages 1–8. IEEE, 2013a. — Cited on page 31.

Fabien Ringeval, Andreas Sonderegger, Juergen Sauer, and Denis Lalanne. Introducing the recola multimodal corpus of remote collaborative and affective interactions. In *2013 10th IEEE international conference and workshops on automatic face and gesture recognition (FG)*, pages 1–8. IEEE, 2013b. — Cited on pages 18, 21, and 22.

Shaghayegh Roohi, Jari Takatalo, J Matias Kivikangas, and Perttu Hämäläinen. Neural network based facial expression analysis of game events: a cautionary tale. In *Proceedings of the Annual Symposium on Computer-Human Interaction in Play (CHI Play)*, pages 429–437, 2018. — Cited on pages 12, 13, and 16.

Julian Runge, Peng Gao, Florent Garcin, and Boi Faltings. Churn prediction for high-value players in casual social games. In *Proceedings of the Conference on Computational Intelligence and Games (CIG)*, pages 1–8, 2014. — Cited on page 14.

James A Russell. A circumplex model of affect. *Journal of personality and social psychology*, 39(6):1161, 1980. — Cited on pages 1, 12, 13, and 15.

Richard M Ryan, Valerie Mims, and Richard Koestner. Relation of reward contingency and interpersonal context to intrinsic motivation: A review and test using cognitive evaluation theory. *Journal of personality and Social Psychology*, 45(4), 1983. — Cited on page 16.

Richard M Ryan, C Scott Rigby, and Andrew Przybylski. The motivational pull of video games: A self-determination theory approach. *Motivation and Emotion*, 30(4):344–360, 2006. — Cited on page 16.

Sara M Schaafsma, Donald W Pfaff, Robert P Spunt, and Ralph Adolphs. Deconstructing and reconstructing theory of mind. *Trends in cognitive sciences*, 19(2):65–72, 2015. — Cited on page 7.

Tom Schaul. A video game description language for model-based or interactive learning. In *Proceedings of the IEEE Conference on Computational Inteligence in Games (CIG)*, pages 1–8. IEEE, 2013. — Cited on page 15.

Ulrich Schimmack and Alexander Grob. Dimensional models of core affect: A quantitative comparison by means of structural equation modeling. *European Journal of Personality*, 14(4):325–345, 2000. — Cited on page 13.

Harold Schlosberg. Three dimensions of emotion. *Psychological review*, 61(2):81, 1954. — Cited on page 12.

Jeff Seger and Richard Potts. Personality correlates of psychological flow states in videogame play. *Current Psychology*, 31(2):103–121, 2012. — Cited on pages 13 and 45.

Rebecca Sevin and Whitney DeCamp. Video game genres and advancing quantitative video game research with the genre diversity score. *The Computer Games Journal*, pages 1–20, 2020. — Cited on page 46.

Ben Seymour and Samuel M McClure. Anchors, scales and the relative coding of value in the brain. *Current Opinion in Neurobiology*, 18(2):173–178, 2008. — Cited on pages 16, 17, 19, and 52.

Noor Shaker and Mohamed Abou-Zleikha. Transfer learning for cross-game prediction of player experience. In *Proceedings of the IEEE Conference on Computational Intelligence and Games (CIG)*, pages 1–8. IEEE, 2016. — Cited on pages 2, 14, 15, 43, and 73.

Noor Shaker and Mohammad Shaker. Towards understanding the nonverbal signatures of engagement in super mario bros. In *International Conference on User Modeling, Adaptation, and Personalization*, pages 423–434. Springer, 2014. — Cited on pages 14 and 17.

Noor Shaker, Georgios Yannakakis, and Julian Togelius. Towards automatic personalized content generation for platform games. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 5, 2010. — Cited on pages 12 and 45.

Noor Shaker, Stylianos Asteriadis, Georgios N Yannakakis, and Kostas Karpouzis. Fusing visual and behavioral cues for modeling user experience in games. *IEEE Transactions on Cybernetics*, 43(6):1519–1531, 2013. — Cited on pages 12, 14, 17, and 24.

Noor Shaker, Mohammad Shaker, and Mohamed Abou-Zleikha. Towards generic models of player experience. In *Proceedings of the Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*, 2015. — Cited on pages 2, 14, 15, 43, and 73.

Karan Sharma, Claudio Castellini, Egon L van den Broek, Alin Albu-Schaeffer, and Friedhelm Schwenker. A dataset of continuous affect annotations and physiological signals for emotion analysis. *Scientific data*, 6(1):1–13, 2019. — Cited on page 19.

Sam Snodgrass, Omid Mohaddesi, Jack Hart, Guillermo Romera Rodriguez, Christoffer Holmgård, and Casper Harteveld. Like peas in pods: the player, environment, agents, system framework for the personalization of digital systems. In *Proceedings of the International Conference on the Foundations of Digital Games (FDG)*, pages 1–15, 2019a. — Cited on pages 14 and 15.

Sam Snodgrass, Omid Mohaddesi, and Casper Harteveld. Towards a generalized player model through the peas framework. In *Proceedings of the International Conference on the Foundations of Digital Games (FDG)*, pages 1–7, 2019b. — Cited on pages 15 and 73.

Richard L Solomon and John D Corbit. An opponent-process theory of motivation: I. temporal dynamics of affect. *Psychological Review*, 81(2):119, 1974. — Cited on pages 17, 19, and 52.

Stanley Smith Stevens et al. On the theory of scales of measurement. *Science*, 1946. — Cited on page 19.

Penelope Sweetser and Peta Wyeth. Gameflow: a model for evaluating player enjoyment in games. *Computers in Entertainment (CIE)*, 3(3):3–3, 2005. — Cited on page 15.

Julian Togelius and Georgios N Yannakakis. General general game ai. In *2016 IEEE Conference on Computational Intelligence and Games (CIG)*, pages 1–8. IEEE, 2016. — Cited on pages 1, 14, 43, 45, 73, and 83.

Gustavo F Tondello, Rina R Wehbe, Lisa Diamond, Marc Busch, Andrzej Marczewski, and Lennart E Nacke. The gamification user types hexad scale. In *Proceedings of the Annual Symposium on Computer-Human Interaction in Play (CHI Play)*, pages 229–243. ACM, 2016. — Cited on page 16.

Ruben Rodriguez Torrado, Philip Bontrager, Julian Togelius, Jialin Liu, and Diego Perez-Liebana. Deep reinforcement learning for general video game ai. In *2018 IEEE Conference on Computational Intelligence and Games (CIG)*, pages 1–8. IEEE, 2018. — Cited on pages 43 and 45.

Amos Tversky and Daniel Kahneman. The framing of decisions and the psychology of choice. *Science*, 211(4481):453–458, 1981. — Cited on page 19.

Greet Van de Perre, Hoang-Long Cao, Albert De Beir, Pablo Gómez Esteban, Dirk Lefeber, and Bram Vanderborght. Generic method for generating blended gestures and affective functional behaviors for social robots. *Autonomous Robots*, 42(3):569–580, 2018. — Cited on page 106.

Laurens van der Maaten and Geoffrey Hinton. Viualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 11 2008. — Cited on page 57.

Vladimir Vapnik and Akshay Vashist. A new learning paradigm: Learning using privileged information. *Neural networks*, 22(5-6):544–557, 2009. — Cited on page 51.

Juan J Vargas-Iglesias. Making sense of genre: The logic of video game genre organization. *Games and Culture*, 15(2):158–178, 2020. — Cited on page 46.

Markus Viljanen, Antti Airola, Jukka Heikkonen, and Tapio Pahikkala. Playtime measurement with survival analysis. *IEEE Transactions on Games*, 10(2):128–138, 2018. — Cited on pages 14 and 44.

Andrej Vítek. Cross-game modeling of player's behaviour in free-to-play games. In *Proceedings of the ACM Conference on User Modeling, Adaptation and Personalization*, pages 384–387, 2020. — Cited on page 14.

Guenter Wallner. Sequential analysis of player behavior. In *Proceedings of the Annual Symposium on Computer-Human Interaction in Play*, pages 349–358. ACM, 2015. — Cited on page 13.

Qishen Wang, Ou Wu, Weiming Hu, Jinfeng Yang, and Wanqing Li. Ranking social emotions by learning listwise preference. In *The First Asian Conference on Pattern Recognition*, pages 164–168. IEEE, 2011. — Cited on page 25.

Chris Westbury, Jeff Keith, Benny B Briesemeister, Markus J Hofmann, and Arthur M Jacobs. Avoid violence, rioting, and outrage; approach celebration, delight, and strength: Using large text corpora to compute valence, arousal, and the basic emotions. *The Quarterly Journal of Experimental Psychology*, 68(8):1599–1622, 2015. — Cited on page 12.

Peter Wittenburg, Hennie Brugman, Albert Russel, Alex Klassmann, and Han Sloetjes. Elan: a professional framework for multimodality research. In *5th International Conference on Language Resources and Evaluation (LREC 2006)*, pages 1556–1559, 2006. — Cited on pages 17 and 18.

Chih-Hung Wu, Yueh-Min Huang, and Jan-Pan Hwang. Review of affective computing in education/learning: Trends and challenges. *British Journal of Educational Technology*, 47(6):1304–1323, 2016. — Cited on page 106.

Yuxia Wu, Ke Li, Guoshuai Zhao, and QIAN Xueming. Personalized long-and short-term preference learning for next poi recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 2020. — Cited on page 105.

Jun Xu and Hang Li. Adarank: a boosting algorithm for information retrieval. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 391–398, 2007. — Cited on page 25.

Tong Xue, Abdallah El Ali, Tianyi Zhang, Gangyi Ding, and Pablo Cesar. Rcea-360vr: Real-time, continuous emotion annotation in 360° vr videos for collecting precise viewport-dependent ground truth labels. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pages 1–15, 2021. URL `https://doi.org/10.1145/3411764.3445487`. — Cited on pages 18 and 20.

Elaheh Yadegaridehkordi, Nurul Fazmidar Binti Mohd Noor, Mohamad Nizam Bin Ayub, Hannyzzura Binti Affal, and Nornazlita Binti Hussin. Affective computing in education: A systematic review and future research. *Computers & Education*, 142:103649, 2019. — Cited on page 106.

Wenlu Yang, Maria Rifqi, Christophe Marsala, and Andrea Pinna. Physiological-based emotion detection and recognition in a video game context. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2018. — Cited on pages 12, 16, 21, and 22.

Georgios N Yannakakis. Enhancing health care via affective computing. *Malta Journal of Health Sciences*, 2018. — Cited on page 106.

Georgios N Yannakakis and John Hallam. Real-time game adaptation for optimizing player satisfaction. *IEEE Transactions on Computational Intelligence and AI in Games*, 1(2): 121–133, 2009. — Cited on page 45.

Georgios N Yannakakis and Hector P Martinez. Grounding truth via ordinal annotation. In *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 574–580, 2015. — Cited on pages 17, 18, 19, 20, 24, 31, 38, 40, and 63.

Georgios N Yannakakis and Héctor P Martínez. Ratings are overrated! *Frontiers in Information and Communication Technology*, 2:13, 2015. — Cited on pages 16, 17, 19, 24, 41, and 63.

Georgios N Yannakakis and Julian Togelius. *Artificial Intelligence and Games*. Springer, 2018. — Cited on pages 1, 2, 11, 13, 14, 23, 32, 45, 46, 73, and 105.

Georgios N Yannakakis, Héctor P Martínez, and Arnav Jhala. Towards affective camera control in games. *User Modeling and User-Adapted Interaction*, 20(4):313–340, 2010. — Cited on pages 21, 22, and 46.

Georgios N Yannakakis, Hector P Martinez, and Maurizio Garbarino. Psychophysiology in games. In *Emotion in games*, pages 119–137. Springer, 2016. — Cited on page 16.

Georgios N Yannakakis, Roddy Cowie, and Carlos Busso. The ordinal nature of emotions. In *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 248–255. IEEE, 2017. — Cited on pages 16, 17, and 44.

Georgios N Yannakakis, Roddy Cowie, and Carlos Busso. The ordinal nature of emotions: An emerging approach. *IEEE Transactions on Affective Computing*, 2018. — Cited on pages 4, 9, 14, 16, 17, 19, 23, 24, 37, 38, 41, 44, 52, 59, and 63.

Chloe Shu-Hua Yeh. Exploring the effects of videogame play on creativity performance and emotional responses. *Computers in Human Behavior*, 53:396–407, 2015. — Cited on page 45.

Stefanos Zafeiriou, Dimitrios Kollias, Mihalis A Nicolaou, Athanasios Papaioannou, Guoying Zhao, and Irene Kotsia. Aff-wild: Valence and arousal'in-the-wild'challenge. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 34–41, 2017. — Cited on page 21.

Tianyi Zhang, Abdallah El Ali, Chen Wang, Alan Hanjalic, and Pablo Cesar. Rcea: Real-time, continuous emotion annotation for collecting precise mobile video ground truth labels. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pages 1–15, 2020. — Cited on pages 18, 19, and 20.